PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-049484
(43)Date of publication of application : 15.02.2002

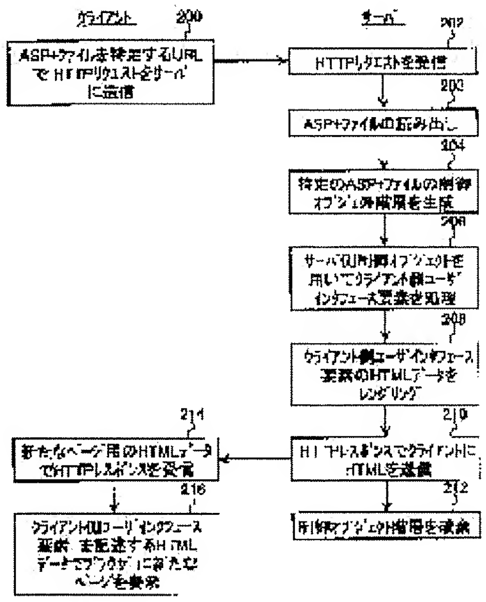
(51)Int.Cl. G06F 9/44
G06F 13/00

(21)Application number : 2001-129923
(22)Date of filing : 26.04.2001
(71)Applicant : MICROSOFT CORP
(72)Inventor : BURD GARY S
COOPER KENNETH B
GUTHRIE SCOTT D
EBBO DAVID S
ANDERS MARK T
PETERS TED A
MILLET STEPHEN J

(30)Priority
Priority number : 2000 573769 Priority date : 18.05.2000 Priority country : US

(54) SERVER SIDE CONTROL OBJECT FOR PROCESSING CLIENT SIDE USER INTERFACE ELEMENT

(57)Abstract:
PROBLEM TO BE SOLVED: To suitably constitute programming required to process a user interface element as a capsule.
SOLUTION: Server side control objects are used for generating the hierarchy of server side control objects in cooperation with a result authoring language code such as the HTML for displaying a client's web page by combining many server side control objects in order to process and generate a client side user interface element to be displayed on the web page. The processing of the client side user interface element is performed by performing one or more processing out of event handling processing, post back data handling


クライアント
200
ASP+ファイルを特定するURLでHTTPリクエストをサーバに送信
サーバ
202
HTTPリクエストを受信
203
ASP+ファイルの読み出し
204
特定のASP+ファイルの制御オブジェクト階層を生成
206
サーバ側制御オブジェクトを用いてクライアント側ユーザインタフェース要素を処理
208
クライアント側ユーザインタフェース要素のHTMLデータをレンダリング
210
HTTPレスポンスでクライアントにHTMLを送信
212
制御オブジェクト階層を破棄
214
新たなページ用のHTMLデータでHTTPレスポンスを受信
216
クライアント側ユーザインタフェース要素を記述するHTMLデータでブラウザに新たなページを表示

processing, data connection processing, and state management processing concerned with the states of the server side control objects.

LEGAL STATUS

[Date of request for examination]	04.12.2003
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	
[Date of registration]	
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

CLAIMS

[Claim(s)]
[Claim 1] The process which receives the request which is the art of the client side user interface element built into the web page displayed on a client, and refers to a resource, The process which reads declaration from said resource, and the process which generates the server side control object programmed based on said declaration to offer the function of said client side user interface element, The process which processes said client side user interface element using said server side control object, The art of a client side user interface element including the process which generates the authoring language data for building said client side user interface element into said web page from said server side control object following on said down stream processing.
[Claim 2] The art of the client side user interface element according to claim 1 said whose resource is the server side declaration data storage section.
[Claim 3] The art of the client side user interface element according to claim 1 which includes further the process which transmits said authoring language data to said client, and the process which terminates said server side control object following on the process which generates said authoring language data.
[Claim 4] Said down stream processing is the art of a client side user interface element including the process which handles the postback data received from said user interface element according to claim 1 using said server side control object.
[Claim 5] Said down stream processing is the art of a client side user interface element including the process which handles the postback event which received from said user interface element according to claim 1 using said server side control object.
[Claim 6] Said down stream processing is the art of a client side user interface element including the process which saves the view state of said server side control object, and the process which transmits said view state to said client according to claim 1.
[Claim 7] Said down stream processing is the art of the client side user interface element according to claim 1 which is the view state of said server side control object, and includes the process which receives the view state corresponding to the condition of said server side control object of a previous request, and the process which loads said view state to said server side control object from said client.
[Claim 8] Shape is taken by the computer system by the subcarrier. It is the computer data signal which is coding the computer program which carries out executive operation of the computer process which processes the client side user interface element built into the web page displayed on a client. And said computer process The process which receives the request which refers to a resource, and the process which reads declaration from said resource, The process which generates the server side control object programmed based on said declaration to offer the function of said client side user interface element, The process which processes said client side user interface element using said server side control object, A computer data signal including the process which generates the authoring language data for building said client side user interface element into said web page from said server side control object following on said down stream processing.
[Claim 9] Reading [computer system] is possible. It is the computer program storage which is coding

the computer program which carries out executive operation of the computer process which processes the client side user interface element built into the web page displayed on a client. Said computer process The process which receives the request which refers to a resource, and the process which reads declaration from said resource, The process which generates the server side control object programmed based on said declaration to offer the function of said client side user interface element, The process which processes said client side user interface element using said server side control object, A computer program storage including the process which generates the authoring language data for building said client side user interface element into said web page from said server side control object following on said down stream processing.
[Claim 10] Process one or more client side user interface elements built into the web page displayed on a client. He is the hierarchy of the server side control object in which executive operation is possible by computer. The hierarchy of said server side control object They are one or more server side child objects corresponding to said one or more client side user interface elements. Each server side child object The postback input received from the client is handled. And one or more server side child objects which generate the authoring language data for displaying a client side user interface element on a client, At least one hierarchy identifier which received from the client relevant to the postback input which specifies one of said server side child objects, The hierarchy of the server side control object containing the server side page object which receives the postback input which it is related with said one or more server side child objects hierarchical, and is distributed to one of said server side child objects based on said hierarchy identifier.
[Claim 11] The hierarchy of the server side control object according to claim 10 in which said page object contains said one or more server side child objects hierarchical.
[Claim 12] The hierarchy of the server side control object according to claim 10 which continues existing until said page object answers a request from a client, and is created and said authoring language data of said web page are generated.
[Claim 13] The hierarchy of the server side control object according to claim 10 which continues existing until said authoring language data of the client side user interface element with which each server side child object answers access to the server side control object specified by said hierarchy identifier, and is created, and it corresponds on a client are generated.
[Claim 14] Reading [computer system] is possible. It is the computer program storage which is coding the computer program which carries out executive operation of the computer process which processes one or more client side user interface elements built into the web page displayed on a client. The process as which said computer process inputs one or more declarations from the server side declaration data storage section, The process which generates the hierarchy of the server side control object programmed based on said declaration to offer the function of said client side user interface element, The process which processes said client side user interface element using the hierarchy of said server side control object, A computer program storage including the process which generates the authoring language data for building said client side user interface element into said web page from the hierarchy of said server side control object.
[Claim 15] The generation process of the hierarchy of said server side control object is a computer program storage including the process which generates the server side container control object corresponding to the client side user interface container element on said web page, and the process which generates one or more server side child control objects corresponding to one or more client side user interface child elements contained in said client side user interface container element according to claim 14.
[Claim 16] Down stream processing of said client side user interface element The process which receives the hierarchy identifier of the meaning which refers to one or more server side control objects in the hierarchy of said server side control object, The process which receives the postback input relevant to the hierarchy identifier of said meaning, The process which decomposes the hierarchy identifier of said meaning in order to identify the server side control object referred to, A computer program storage including the process which passes said postback input to said server side control object

referred to, and the process which processes said postback input using said server side control object referred to according to claim 15.
[Claim 17] Down stream processing of said client side user interface element is a computer program storage including the process which traverses the hierarchy of said server side control object in order to call each processing by one or more server side control objects according to claim 14.
[Claim 18] Down stream processing of said client side user interface element The process which registers one or more server side control objects in order to handle the control object event generated by one of said server side control objects, The process which generates said control object event from said one or more server side control objects, A computer program storage including the process which handles said control object event using said one or more server side control objects registered in order to handle said control object event according to claim 14.
[Claim 19] Down stream processing of said client side user interface element The process which loads the 1st view state to said one or more server side control objects, The process which handles postback data using said one or more server side control objects following on the load process of said 1st view state, The process which handles a postback event using said one or more server side control objects following on the process which handles said postback data, A computer program storage including the process which continues at the process which handles said postback event, and saves the 2nd view state from said one or more server side control objects according to claim 14.
[Claim 20] Down stream processing of said client side user interface element is a computer program storage according to claim 19 which includes further the process which solves the data-coupling relation between a server side control object and the server side data storage section following on the process which handles a postback event.
[Claim 21] The process which is the art of at least one client side user interface element built into the web page displayed on a client, and reads declaration from a resource, The process which generates two or more server side control objects corresponding to said client side user interface element and logic target which recognize simultaneous existence based on said declaration, The process which processes a client side user interface element using said server side control object which recognizes simultaneous existence, The art of a client side user interface element including the process which generates the authoring language data for building said client side user interface element into said web page from said server side control object which recognizes simultaneous existence following on said down stream processing.
[Claim 22] The generation process of the server side control object of said plurality which recognizes simultaneous existence is the art of the client side user interface element according to claim 21 which includes the process which generates the 2nd server side control object corresponding to the 1st client side user interface element so that the process which generates the 1st server side control object corresponding to the 1st client side user interface element, said 1st server side control object, and said 2nd server side control object may exist simultaneously.
[Claim 23] The art of a client side user interface element including the process at which said down stream processing starts the server side event relevant to the 1st server side control object according to claim 21.
[Claim 24] The art of the client side user interface element according to claim 23 with which said down stream processing includes further the process which handles said server side event using the 2nd server side control object.
[Claim 25] The art of the client side user interface element according to claim 23 with which said down stream processing includes further the process which handles a server side event using a non-user interface server component.
[Claim 26] The art of a client side user interface element including the process which generates the 2nd server side control object corresponding to said client side user interface element so that the process at which the generation process of the server side control object of said plurality which recognizes simultaneous existence generates the 1st server side control object corresponding to a client side user interface element, said 1st server side control object, and said 2nd server side control object may exist

simultaneously according to claim 21.
[Claim 27] Shape is taken by the computer system by the subcarrier. It is the computer data signal which is coding the computer program which carries out executive operation of the computer process which processes at least one client side user interface element built into the web page displayed on a client. The process to which said computer process reads declaration from a resource, and the process which generates two or more server side control objects corresponding to said client side user interface element and logic target based on said declaration to which simultaneous existence is recognized, The process which processes a client side user interface element using said server side control object which recognizes simultaneous existence, A computer data signal including the process which generates the authoring language data for building said client side user interface element into said web page from said server side control object which recognizes simultaneous existence following on said down stream processing.
[Claim 28] Reading [computer system] is possible. It is the computer program storage which is coding the computer program which carries out executive operation of the computer process which processes at least one client side user interface element built into the web page displayed on a client. The process to which said computer process reads declaration from a resource, and the process which generates two or more server side control objects corresponding to said client side user interface element and logic target based on said declaration to which simultaneous existence is recognized, The process which processes a client side user interface element using said server side control object which recognizes simultaneous existence, A computer program storage including the process which generates the authoring language data for building said client side user interface element into said web page from said server side control object which recognizes simultaneous existence following on said down stream processing.

[Translation done.]

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] Generally especially this invention relates to the server side control object which processes the client side user interface element of a web page about a web server framework.
[0002]
[Description of the Prior Art] A typical web browser receives data from the web server which defines the appearance of a web page and basic actuation which are displayed in a client system. As a typical procedure, a user specifies the uniform resource (resource) locator (henceforth "URL") which is the global address of the resource on World Wide Web, and a desired website is accessed. Generally, the vocabulary "a resource" is the data which can be accessed by the program, or a routine.
[0003] An example of URL is "HYPERLINK "http://www.microsoft.com/ms.htm" http://www.microsoft.com/ms.htm." The 1st part of this example of URL shows the given protocol (for example, "http") used for a communication link. The 2nd part specifies the domain name (for example, "HYPERLINK "http://www.microsoft.com" www.microsoft.com") which shows the location of a resource. The 3rd part specifies the resource in a domain (for example, file called "ms.htm"). This is followed and they are a browser and HYPERLINK"http://www.microsoft.com" www.microsoft.com. The HTTP (hypertext transport protocol) request relevant to the example of URL for taking out the data relevant to the ms.htm file in a domain is generated. The web server which is acting as the host of the www.microsoft.com site receives a HTTP request, returns the demanded web page or resource to a client system by the HTTP response, and displays it on a browser.
[0004] The "ms.htm" file of the above-mentioned example contains the static HTML (HyperText Markup Language) code. HTML is a plane (plaintext) text authoring language used for the document (for example, web page) creation on World Wide Web. Since it is such, an HTML file can be displayed on a browser as a web page, in order to offer graphical experience which a user expects, while being taken out from a web server and displaying the information from the Internet.
[0005] A developer can specify the text and list which are displayed on a browser and by which formatting was carried out, form, a table, a hypertext link, an in-line image, voice, and background graphics using HTML. However, an HTML file is a static file which is not supporting dynamic generation of web page contents in essence.
[0006] Moreover, a web page may need to display dynamic contents, such as change, traffic information, etc. on a stock price, on a browser. In such a situation, a typical server side application program acquires dynamic data, and it is developed so that it may be formatted into the HTML format transmitted to the browser displayed on a web page while a web page is updated.
[0007] Furthermore, although data are not strictly dynamic, since the value displayed on a static web page is a value from which many differ dramatically, in a situation it is not practical to create a demanded number of static web pages, these same servers side application program can be used. For example, a travel plan page may give two kinds of calender indication with the calender for start days, and the calender for return days. Instead of developing the static page of hundreds by the combination of

all the calenders that can be considered, a server side application program can generate dynamically the suitable static page which displayed the suitable calender.
[0008] By many web pages, a user can have a dialog with the page displayed on the browser by choosing the visual element of a page. For example, in the above-mentioned travel plan page, with a calender, or a user clicks the date and chooses the date, he can have a dialog with the calender by clicking on an icon, carrying forward the moon or returning. A browser advances a HTTP request to a server side application program by the existing solution. This HTTP request can contain the parameter coded in addition to this in an enquiry matrix like a form post variable, or the data format which describes a client side event or data (for example, which control did the user click?). For example, a parameter may contain the data which the user chose in one calender with data current on display in the calender of another side.
[0009] The communication link of the event and data which are returned to a server is called the "postback." This is because a browser transmits a request using a HTTP post request typically. A server side application program processes a HTTP request, and generates the suitable HTML code for the web page which has the newly calculated calender reflecting action of the user who transmits to a client by the HTTP response. Then, the obtained document is transmitted to a client system by the HTTP response, and this document is displayed on a browser as a web page which shows the updated calender.

[0010] That it is well versed in the pro GURAMUBE six which data are transmitted between a browser and a server how development of a server side application program is not only well versed in the usual HTML coding used for a web page design, but, or contains one or more programming language (for example, C++, Perl, Visual Basic, or Jscript) and HTTP protocols is the complicated activity demanded. However, a web page architect is graphic designer or an editor in many cases, and may be inexperienced in a program. Furthermore, if complicated web page development is simplified, the rate of development of new web contents can be gathered by any developers.
[0011] Generally, great efforts are required and development of a custom-made server side application program is also actually like [which a developer regards as not wanting to often try it]. In order to display a desired web page, a developer not only understands the HTML code as which generation is required, but has to understand what the user dialogue and client data from a web page become postback processing. Therefore, it is desirable to offer the development framework to which a developer can create and process a web page dynamically by the minimum programming.
[0012]
[Problem(s) to be Solved by the Invention] One means which makes min the requirements for a program of dynamic web page generation is an active server page (ASP) framework offered by Microsoft Corp. An ASP resource processes typically the HTTP request which specifies an ASP resource as a desired resource, after that, generates the HTML code as a result of the HTTP response to a client, for example, contains Visual Basic or Jscript. Furthermore, in order to ease given application programming efforts, or the ASP resource was developed beforehand, refer to a third party's client side library component (for example, client side "ACTIVEX" control) for it. However, in the present server side application framework, programming which must manage dynamically client side user interface elements (for example, a text box, a list box, a carbon button, a hyperlink, an image, voice, etc.) within server side application needs a still advanced programming technique and considerable efforts. An unsolved technical problem is about things for which a user interface element is processed, such as handling a postback event, encapsulating programming demanded appropriately so that a web page developer can focus on other aspects of a web page.
[0013]
[Means for Solving the Problem] According to this invention, **** and other technical problems are solved by offering the server side control object framework which manages processing and generation of a client side user interface element. Furthermore, the hierarchy of a server side control object can collaborate and generate an authoring language code a result like the criterion HTML for displaying a web page on a client. As long as a client supports an authoring language code as a result of [another]

for example, the criterion HTML, it may be what kind of browser. Processing of a client side user interface element includes one or more postback event handling processings, postback data handling processing, data-coupling processing, or the status management processing about the condition of a server side control object.
[0014] The big effectiveness of an operation gestalt with this invention is in improvement in capsulation of server side processing of a client side user interface element (for example, output used for the reception input from a user interface element, and generation of a user interface element), and related functionality. One or more server side control objects may be generated so that it may correspond to one or more user interface elements logically. For example, considering the user interface element which expresses the moon in a KARENTA display, the hierarchy of a server side control object may be generated corresponding to a calender display and its various sub element. In the "moon" control object, with a certain configuration, the ** "week" control object contains seven "day" control objects hierarchical, including many "week" control objects hierarchical.
[0015] Furthermore, in an operation gestalt with this invention, a server side control object can collaborate and process the user interface element which corresponds logically. This advantage is the result of being brought when many server side control objects exist simultaneously during client request processing and generation of a response. For example, if the 1st postback event (for example, click of the carbon button element of the "next moon" on a calender display) which received from the client is detected, one control object will start the 2nd event (for example, the selection and the display of the next moon by the "moon" control object) which is detected after that and processed by one or more control objects which are recognizing simultaneous existence. By this collaboration, a complicated control dialogue can be encapsulated within the server side control object itself, and, thereby, the custom-made event handling required of a web page developer can be made into the minimum.
[0016] The approach and computer program product which process the client side user interface element built into the web page displayed on a client are offered. The request which refers to the server side resource data storage section is received. Declaration is inputted from the server side declaration data storage section. A server side control object is generated and programmed to give the functionality of a user interface element based on declaration. A user interface element is processed using a server side control object. Authoring language data are generated from the server side control object which displays a user interface element on a web page.
[0017] The hierarchy of the server side control object in which executive operation is possible is offered by computer which processes one or more client side user interface elements built into the web page displayed on a client. One or more server side child objects correspond to one or more client side user interface elements. Each server side child object treats the input received from the client, and generates the authoring language data for displaying a client side user interface element on a client. At least one hierarchy identifier is received from the client relevant to the input which specifies one of server side child objects. A server side page object receives the input distributed to one of the server side child objects according to a hierarchy identifier including a server side child object.
[0018]
[Embodiment of the Invention] The operation gestalt with this invention contains the server side control object which processes and generates the client side user interface element displayed on a web page. Furthermore, the hierarchy of a server side control object can collaborate and generate an authoring language code a result like the criterion HTML for displaying a web page on a client. As long as a client supports an authoring language code as a result of [another] for example, the criterion HTML, it may be what kind of browser. With an operation gestalt with this invention, a server side control object corresponds to a client side user interface element and a logic target, and generates the authoring language code used by the client side browser which displays and processes a web page by the server. Processing of a client side user interface element may also include one or more postback event handling processings, postback data handling processing, data-coupling processing, and status management processing.
[0019] Drawing 1 shows the web server which generates dynamically the web page contents displayed

on the client in an operation gestalt with this invention. A client 100 performs the browser 102 which displays the web page 104 on the indicating equipment of a client 100. A client 100 may also include the client computer system which has a display like a video monitor. The "INTERNET EXPLORER" browser currently sold by Microsoft Corp. is an example of the browser 102 in an operation gestalt with this invention. As an example of other browsers, it is "NETSCAPE NAVIGATOR". It reaches. Although there is "MOSAIC" etc., it does not restrict to this. The text box control 106 and two carbon button control 108 and 110 are included in the illustrated web page 104. A browser 102 can receive the HTML code from a web server 116 by the HTTP response 112, and displays the web page described by the HTML code. Although HTML is explained with reference to a certain operation gestalt Not the thing restricted especially but SGML (Standard Generalized Markup Language), XML (eXtensible Markup Language) -- and It is the markup language of the XML base. It is thought that other authoring languages containing WML (Wireless Markup Language) designed so that the content and user interfaces of narrow-band radio equipment, such as a pocket bell (trademark) and a cellular phone, might be specified are within the limits of this invention. Furthermore, although the criterion HTML 3.2 is mainly indicated in this description, any versions of HTML may be contained within the limits of this invention.
[0020] The communication link with a client 100 and a web server 116 can be performed using a series of processings of the HTTP request 114 and the HTTP response 112. Although HTTP is explained with reference to a certain operation gestalt, it is not restricted especially and it is thought that other transport protocols including S-HTTP are within the limits of this invention. In a web server 116, the HTTP pipeline module 118 receives the HTTP request 114, analyzes URL, and calls the suitable handler which processes a request. The web server 116 is equipped with two or more handlers 120 which handle the resource of a different type in the operation gestalt with this invention.
[0021] For example, when URL specifies a static contents resource 122 like an HTML file, a handler 120 accesses the static contents resource 122, and sends the static contents resource 122 to a client 100 by the HTTP response 112 through the HTTP pipeline 118. Or in an operation gestalt with this invention, when URL specifies a dynamic contents resource 124 like an ASP+ resource, a handler 120 accesses the dynamic contents resource 124, processes the contents of the dynamic contents resource 124, and generates the HTML code the result for web page 104. In an operation gestalt with this invention, the result HTML code contains standard HTML3.2 code. Generally, a dynamic contents resource is the server side declaration data storage section (for example, ASP+ resource) which can use the authoring language which describes the web page which should be displayed on a client for generating dynamically. And the HTML code for web pages passes the HTTP pipeline 118, and is sent to a client 100 by the HTTP response 112.
[0022] During this processing, the handler 120 was beforehand developed again, in order to simplify a development effort, or it can access the library of the 3rd person code. One of such the libraries is the server side class control library 126, and a handler 120 can illustrate from here the server side control object which generates HTML data as a result of processing a user interface element and displaying on a web page. in an operation gestalt with this invention, one or more server side control objects are visible on the web page described by the dynamic contents resource 124 -- it hides-like and maps to one or more user interface elements.
[0023] On the other hand, the 2nd library is a client side control class library 128 like the library containing the "ACTIVEX" component from Microsoft Corp. "ACTIVEX" control is a COM (component object model) object which follows a fixed criterion in the method of a dialogue with a client and other components. Client side "ACTIVEX" control is the component of the COM base as for which downloads to a client automatically and executive operation may be carried out to it by the web browser of a client. A server side ACTIVEX component (not shown) is a component of the COM base which may be performed on a server, in order to achieve the various server side functions in which the server side functionality of stock price retrieval application or a database component is offered. ACTIVEX is indicated by the detail by "understanding of ACTIVEX and OLE" (David Chappelle, the Microsoft press, 1996).

[0024] In contrast with "ACTIVEX" control, the server side control object of the operation gestalt of this invention specified as the dynamic contents resource 124 corresponds to the user interface element with which it is displayed on a client logically. A server side control object can generate the effective HTML code which may contain the locator which refers to for example, a HTML tag and given client side "ACTIVEX" control again. When the browser has already had the code for client side "ACTIVEX" control in the storing system, executive operation of the "ACTIVEX" control is carried out within the web page on a client. If it becomes right [that], a browser will download the code for "ACTIVEX" control from the resource specified by the locator, and will carry out executive operation of the "ACTIVEX" control within the web page on a client. The server side control object in the operation gestalt of this invention can start an event again to the server side "ACTIVEX" object used for performing stock price retrieval application on a server.
[0025] A handler 120 accesses one or more non-user interface server components 130 which carry out executive operation on a web server 116 or another accessible web server again. A non-user interface server component 130 like stock price retrieval application or a database component is referred to in the dynamic contents resource 124 processed by the handler 120, or is related with it. The server side event started by the control object declared with the dynamic contents resource 124 may be processed in server side code which calls the suitable method of the non-user interface server component 130. Consequently, the processing offered by the server side control object can simplify programming of non-user interface server component stereo-NENTO 130 by encapsulating processing and generation of a web page of a user interface element, and, thereby, the developer of the non-user interface server component 130 can be concentrated on development of the function of the application proper instead of a user interface problem.
[0026] Drawing 2 shows the flow chart of processing of the client side user interface element using the server side control object in an operation gestalt with this invention, and generation processing. In processing 200, a client transmits a HTTP request to a server. A HTTP request contains URL which specifies resources, such as an ASP+ resource. In processing 202, a server calls the suitable handler which receives a HTTP request and processes the specified resource. Reading appearance of the ASP+ resource is carried out in processing 203. Processing 204 generates a server side control object hierarchy based on the content of the specified dynamic contents resource (for example, ASP+ resource).
[0027] In processing 206, a control object hierarchy's server side control object performs postback event handling, postback data handling, status management, and one or more processings in data coupling. From a client, the postback event and data (collecting "postback input") from a user interface element are sent to a server, and are processed. Especially a postback event is not restricted and may also contain the "data change" event from the client side text box element sent to "a mouse click" or a server from a client side carbon button element. Especially postback data may also contain the text in which **** was inputted into the index of the item chosen from for example, the text box element or the drop down box by the user in what is restricted. However, postback processing may be performed by other events and is only performed by the dialogue with a user.
[0028] In processing 208, in order that each server side control object in a hierarchy may generate authoring language data like the HTML code for the display by the web page of a client side user interface element (or rendering), it is called. Although the vocabulary "a rendering" may mean the processing which displays graphics on a user interface, in this description, the vocabulary "a rendering" also means generation processing of the authoring language data which may be interpreted by client application like the browser for a display and a client side function. More detailed explanation of processing 206 and the rendering processing 208 is given in connection with drawing 6 . In a certain operation gestalt, the call of the render() method in each control object is performed using a tree traversal sequence. That is, the call of the render() method of a page object becomes the repetitive traverse covering the suitable server side control object in a hierarchy. As an option which calls the render() method of a suitable control object, approaches, such as event signaling or the object registration technique, may be used. A parenthesis specifies the "render()" level which shows the method of comparing with a data value.

[0029] In an operation gestalt with this invention, actual creation of each server side control object may delay until a server side control object is accessed in processings 206 or 208 (handling of a postback input, loading of a condition, rendering of a control object to the HTML code, etc.). When a server side control object is not accessed for a given request, server processing is optimized by delaying creation of a control object and eliminating unnecessary control object creation processing.
[0030] In processing 210, the HTML code is transmitted to a client by the HTTP response. In processing 214, a client receives the HTML code relevant to the new web page which should be displayed. In processing 216, a client system incorporates the user interface element of a new page according to the HTML code received from the HTTP response (for example, it displays). however, a user interface element -- incorporating -- he should also understand that you may include non-display processing of control of read-out to the memory which offers voice or a tactile-sense output and writing, and script processing etc. A server side control object hierarchy is ended in processing 212. In an operation gestalt with this invention, the server side control object in a hierarchy answers the HTTP request which refers to the ASP+ response of relation, is created, and after the rendering of authoring language data (for example, HTML data) finishes, it is destroyed. In another operation gestalt, processing 212 may be performed before processing 210 after processing 208.
[0031] Drawing 3 shows an example of the module in the web server used in an operation gestalt with this invention. A web server 300 accepts the HTTP request 302 in the HTTP pipeline 304. The HTTP pipeline 304 may also contain various modules, such as logging of web page statistics, user collating, a right of user access, and a module for the formation of an output cache of a web page. Eventually, each input HTTP request 302 received by the web server 300 is processed by the specific instance of an IHTTP handler class (it illustrates as a handler 306). A handler 306 decomposes a URL request and calls a suitable handler factory (for example, page factory module 308).
[0032] In drawing 3 , the page factory 308 related with the ASP+ resource 310 is called, and instantiation and configuration of the ASP+ resource 310 are handled. In a certain operation gestalt, an ASP+ resource may be recognized by specifying a specific suffix (or a file extension part like ".aspx") as a file. If the request to given ".aspx" resource is first received by the page factory module 308, the page factory module 308 will search a file system, and will obtain a suitable file (for example, .aspx file 310). This file may also contain the text (for example, authoring language data) which may be behind interpreted or accessed by the server which processes a request, or the data (for example, cutting tool code data or coded data) in another format. When a physical file exists, the page factory module 308 reads a file to an aperture, and reads the file to memory. When a file is not found, the page factory module 308 returns the suitable error message "a file is not found."
[0033] After reading the ASP+ resource 310 to memory, the page factory module 308 processes a file content, and builds the data models (for example, a list [of script blocks], directive, static text area, and hierarchy server side control object, a server side control property, etc.) of a page. A data model is used for generating the source list of a new object class like the COM+ (Component Object Model+) class to which the class of the page base is made to extend. The page base class contains the code which specifies the structure, property, and function of a basic page object. A source list is dynamically compiled by intermediate language and is behind compiled [in / next / an operation gestalt with this invention] by the instructions (for example, X86, Alpha, etc.) of a proper on a plat form timely (Just-In-Time). Intermediate language is COM+ IL. A code and Java A cutting tool code and Modula 3 A code and SmallTalk A general purpose or custom-made orientation linguistic code of a code, the Visual Basic code, etc. may also be included. In another operation gestalt, intermediate-language processing may be excluded and a native instruction may be directly generated from a source list or a source file (for example, ASP+ resource 310). The control class library 312 may be accessed with the page factory module 308, in order to obtain the server side control class which is used for a control object hierarchy's generation and which was defined beforehand.
[0034] The page factory module 308 outputs the page object 314 which is a server side control object equivalent to the web page 104 of drawing 1 . The page object 314 and its child object (for example, the text box object 318, the carbon button object 320, and another carbon button object 322) are the control

object hierarchy's 316 examples. The example of other control objects can be considered according to this invention, like the custom-made control object, is not restricted especially and also contains the object corresponding to HTML control of a table 1. The page object 314 corresponds to the web page 104 of drawing 1 . The text box object 318 corresponds to the text box 106 of drawing 1 . Similarly, the carbon button object 320 corresponds to the additional carbon button 108 of drawing 1 , and the carbon button object 322 corresponds to the deletion carbon button 110 of drawing 1 . The page object 314 relates to other control objects and hierarchy targets on a server. In a certain operation gestalt, a page object is a container object which contains the child control object hierarchical. The hierarchical relationship of other gestalten, such as a dependency, can be used with another operation gestalt. In a complicated control object hierarchy, one child object may be a container object of other child objects rather than it has the child object of a large number level.
[0035] It sets in the above-mentioned operation gestalt, the control object hierarchy's 316 control object is created and performed on a server 300, and each server side control object corresponds to the corresponding user interface element and corresponding logic target on a client. A server side control object collaborates, handles the postback input from the HTTP request 302 again, manages the condition of a server side control object, performs data coupling with a server side database, and generates the authoring language data (for example, the HTML code) used for the display of a web page as a result of a client. Result authoring language data are generated from the server side control object hierarchy 316 (namely, rendering), and are transmitted to a client by the HTTP response 324. For example, the result HTML code can materialize any effective HTML configurations, and in addition to this, control of an ACTIVEX (trademark) type, a JAVA (trademark) applet, a script, and when it is processed by the browser, refer to any web resources which produce client side user interface elements (for example, a control carbon button, a text box, etc.) for it.
[0036] By declaration made with the ASP+ resource 310, a server side control object can access one or more non-user interface server components 330 for a dialogue with the non-user interface server component 330 and a client side user interface element. For example, a postback input can be answered and a server side control object can start a server side event to the non-user interface server component registered into those events. Thus, the non-user interface server component 330 minds a user interface element for a dialogue with a user, and it can perform it, without programming a code required displaying and processing these elements.
[0037] Drawing 4 shows the content of an example of the dynamic contents resource in an operation gestalt with this invention. In the illustrated example, the file 400 includes plain text declaration in a certain dynamic contents resource format (for example, ASP+). Each declaration gives an instruction to the page compiler which reads a file 400, in order to process a client side user interface element, it performs creation and a call of a suitable server side control object, and it combines the HTML code which is eventually transmitted to a client by the HTTP response and by which the rendering was carried out. And describe or refer to the function of the client side user interface element performed by the server side control object for declaration. And a server side control object generates the HTML code used in order to define the new version of the web page on a client.
[0038] The 1st line of a file 400 is :<%@ directive containing the directive of the following formats. [attribute=value] %> It is here, and especially directive is not restricted and may also contain "page", "cache", or "import". In order to determine buffering semantics, the requirements for a session condition, an error handling scheme, scree PUTINGU language, transaction semantics, and a property like an import directive, a directive is used by the page compiler, when processing a dynamic contents resource. A directive may exist anywhere in a page file.
[0039] <html> of the 2nd line is a standard HTML initiation tag passed to the result HTML code as a literal (that is, additional processing is not performed in order to carry out the rendering of the result HTML code). In HTML functor, the beginning of an HTML file is shown and it has become the termination tag </html> of the line 21 this [whose] is also a literal, and a pair.
[0040] A code declaration block exists in the lines 3-10 of a file 400. Generally, a server side code declaration block defines the method by which executive operation is carried out on a page object, a

control object member variable, and a server. the following formats -- setting -- :<script runat = "server" [language = "language"] and [src = "externalfile"] -- > .... </script> -- here, language and a src parameter are arbitrary. In an operation gestalt with this invention, a code declaration block is defined using the <script> tag including the "runat" attribute which has the value set as the "server." Since the syntax of an inner code is specified, a "language" attribute may be used for arbitration. Although default language can express the language configuration of a whole page, a developer is Jscript by the "language" attribute of a code declaration block. And different language within the same (Practical Extraction and Report Language) web page activation of PERL etc. can be used. The <script> tag can specify a "src" file as arbitration again, and a "src" file is a file of the exterior where a code is inserted in the dynamic contents resource for processing by the page compiler from there. Although the syntax currently indicated is used in this operation gestalt, with another operation gestalt, he should understand that different syntax within the limits of this invention can be used.
[0041] In drawing 4 , two subroutines, AddButton#Click, and DeleteButton#Click are declared in the Visual Basic format into the code declaration block. If any subroutine takes two input parameters, "Source", and "E" and a client side click event is detected by the carbon button of a response, executive operation will be carried out on a server by the response to a HTTP request. In an AddButton#Click subroutine, the text in a user name text box is connected with word "Add", and is loaded to the text data member of a message. In a DeleteButton#Click subroutine, the text in a user name text box is connected with word "Delete", and is loaded to the text data member of a message. Although not shown in drawing 4 , the member variable of a server side control object may be declared with the code declaration block of a file 400. For example, if Visual Basic syntax is used, keyword tooth-space"DIM" will declare the data variable of a server side control object.
[0042] "A code rendering block" (not shown) may be included in a dynamic contents resource. In an operation gestalt with this invention, executive operation of the code rendering block is carried out by one "rendering" method by which executive operation is carried out at the time of a page rendering. A code rendering block fills the following formats (other formats are considered in another operation gestalt).
[0043] <% InlineCode %> "InlineCode" includes the independent language mold code block or flows-of-control block which carries out executive operation on a server at the time of a rendering here.
[0044] An in-line expression can also be used within a code rendering block using the following instantiation-syntax.
[0045] <%= InlineExpression %> Here, the expression included in the "InlineExpression" block is eventually included by the call to "Response.Write (InlineExpression)" of the page object which writes the value from "InlineExpression" in the holder of the suitable location in declaration. For example, "InlineExpression" may be contained in the following code rendering blocks.
[0046] <font size = "<%=x%>" > Hi <%=Name%>, you are <%=Age%>! </font> This outputs a greeting and the description about a certain man's age with the font stored in value"x." The man's identifier and age are defined as a string in a code declaration block (not shown). The rendering of the result HTML code is carried out by the server, and it is transmitted to a client by the HTTP response so that the value of "InlineExpression" may be included in a suitable location. That is, it is as follows.
[0047] <font size ="12"> Hi Bob and you are 35! In the line 11 of a file 400, <body> is a standard HTML tag for specifying the beginning of the text of a HTML document. The termination tag </body> is shown in the line 20 of FIIRU 400. In an operation gestalt with this invention, both <body> and </body> are literals.
[0048] The initiation tag <form> of a HTML foam block is seen on a line 12 in this section of the file 400 of drawing 4 . The termination tag </form> of a foam block is seen on the line 19 of HTML file 400. Since a given identifier is related with a foam block, parameter "id" of arbitration can also be included in a HTML control tag, and it enables this to contain many foam blocks in one HTML file.
[0049] The server side label identified by the "message" is declared in the line 18 of a file 400. A "message" label is used in code declared with the lines 5 and 8 of a file 400, in order to display a label on a web page.

[0050] In the foam block, three examples of a HTML control tag corresponding to the user interface elements 106, 108, and 110 of drawing 1 are shown. The 1st user interface element is declared with the line 13 of the file 400 equivalent to a text box. Text literal "User Name" declares the label located in the left-hand side of a text box. The input tag which has type="Text" declares the text box server side control object which considers as the server side control object which carries out the rendering of the text box client side user interface element, and has an identifier called "UserName". The lines 15 and 16 of a file 400 declare the client side user interface element shown as carbon buttons 108 and 110 of drawing 1 , respectively. A "OnServerClick" parameter specifies the suitable subroutine declared with the code declaration block of a file 400. And the server side carbon button control object generated by the response to declaration by the file 400 carries out the rendering of the server side code of the relation which performs the HTML code and carbon button click event of a client side carbon button.
[0051] The text box and carbon button which were declared by the file 400 are the example of HTML server control declaration. In an initial state, the HTML tags in an ASP+ resource are dealt with [no] as literal text contents, and can be accessed in a page developer's programming. However, when a page developer specifies using the "runat" attribute which has the value set as "server" can show that the syntax of a HTML tag should be analyzed and it should be treated as accessible server control declaration. Each server side control object can be related with the "id" attribute of the meaning which enables program reference of a corresponding control object at arbitration. The property argument on a server side control object and event association can also be specified using the declaration name / value which is an attribute pair in a tag element (for example, OnServerClick is equal to a "MyButton#Click" pair).
[0052] In an operation gestalt with this invention, the general syntax which declares a HTML control object is as follows.
[0053]
<HTMLTag id = "Optional Name" runat = server> .... </HTMLTag> -- here, "Optional Name" is the identifier of the meaning of a server side control object. Although the list, the related syntax, and COM+ class of a HTML tag which are supported now are shown in a table 1, other HTML tags can be considered within the limits of this invention.
[0054]
[A table 1]

HTML タグ名	例	COM+ クラス
`<a>`	`<a id = "MyAnchor" runat = server> My Link </a>`	AnchorButton
`<img>`	`<img id = "MyImage" runat = server>`	Image
`<span>`	`<span id = "MyLabel" runat = server> </span>`	Label
`<div>`	`<div id = "MyDiv" runat = server>Some contents</div>`	Panel
`<form>`	`<form id = "MyForm" runat = server> </form>`	FormControl
`<select>`	`<select id = "MyList" runat = server>` `<option>One</option>` `<option>Two</option>` `<option>Three</option>` `</select>`	DropDownList
`<input type = file>`	`<input id = "MyFile" type = file runat = server>`	FileInput
`<input type = text>`	`<input id = "MyTextBox" type = text>`	TextBox
`<input type = password>`	`<input id = "MyPassword" type = password>`	TextBox
`<input type = reset>`	`<input id = "MyReset" type = reset>`	Button
`<input type = radio>`	`<input id = "MyRadioButton" type = radio runat = server>`	RadioButton
`<input type = checkbox>`	`<input id = "MyCheck" type = checkbox runat = server>`	CheckBox
`<input type = hidden>`	`<input id ="MyHidden" type = hidden runat = server>`	HiddenField
`<input type = image>`	`<input type = image src = "foo.jpg" runat = server>`	ImageButton
`<input type = submit>`	`<input type = submit runat = server>`	Button
`<input type = button>`	`<input type = button runat = server>`	Button
`<button>`	`<button id = MyButton runat = server>`	Button
`<textarea>`	`<textarea id = "MyText" runat = server>` This is some sample text `</textarea>`	TextArea

[0055] In addition to a standard HTML control tag, an operation gestalt with this invention enables a developer to create the reusable component encapsulated about a program function common to the outside of a HTML tag set. These custom-made server side control objects are specified using the declaration tag in a page file. Custom-made server side control object declaration includes the "runat" attribute which has the value set as "server". In order to carry out possible [of the program reference of a custom-made control object], the "id" attribute of a meaning is specified as arbitration. Furthermore, the declaration name / value attribute pair of a tab element specify the property argument of a server side control object, and event association. An in-line template parameter may also be combined with a server side control object by giving a suitable "template" prefix child element to a parent server control object. A format of custom-made server side control object declaration is as follows.
[0056] <servercntrlclassname id="OptionalName" [propertyname="propval"] runat=server/> Here, "servercntrlclassname" is an accessible control class, "OptionalName" is the identifier of the meaning of a server side control object, and "propval" expresses the property value of the arbitration of a control object.
[0057] An XML tag prefix can be used for offering the briefer notation for specifying a server side control object in a page by using the following formats using another specification statement method.
[0058] <tagprefix:classname id = "OptionalName" runat =server/> Here, in "tagprefix", in relation to a given control name tooth-space library, "classname" expresses the identifier of control in a relation name tooth-space library. "propertyvalue" of arbitration is supported.
[0059] In short, the operation gestalt of this invention contains on a server creation and the server side

control object by which executive operation is carried out, in order to generate the HTML code sent to a client. The HTML code can embody any effective HTML configurations, for example, is control of an ACTIVEX type and JAVA. Any web resources which generate the user interface carbon button and other user interface elements in an applet, a script, and other clients can be referred to. The user in a client can have a dialog with these user interface elements that correspond to a server side control object logically, and can return a request to a server. A server side control object is re-created on a server, and it processes the data, the event, and other properties of a user interface element so that the HTML code of the next round which should transmit to a client as a response may be generated.
[0060] If drawing 5 is referred to, an example of the computer system of the operation gestalt of this invention contains the general purpose computer equipment of the gestalt of the conventional computer system 500 containing the system bus 506 which connects the various system components containing the processor unit 502, a system memory 504, and a system memory 504 to the processor unit 500. System buses 506 may be any of the bus structure of some types containing the peripheral bus and local bus which use a memory bus or a memory controller, and various bus architectures. The system memory contains the memory (ROM) 508 only for playbacks, and random access memory (RAM) 510. The unformatted input / output system 512 (BIOS) containing the basic routine which helps a transfer of the information between the elements within a computer system 500 are stored in ROM508.
[0061] The computer system 500 contains the optical disk drive 518 which performs read-out and the writing of a removable optical disk 519 like the magnetic disk drive 514 and CD ROM which perform further read-out and the writing of a hard disk drive 512 and the removable magnetic disk 516 which perform read-out and the writing of a hard disk, DVD, or other optical media. The hard disk drive 512, the magnetic disk drive 514, and the optical disk drive 518 are connected by the hard disk drive interface 520, the magnetic disk drive interface 522, and the optical disk drive interface 524 system bus 506, respectively. A drive and its medium which can be related computer read offer the instruction of a computer system 500 which can be computer read, DS, a program, and the non-volatile storage section of other data.
[0062] Although the hard disk, the removable magnetic disk 516, and the removable optical disk 519 are used for this description in the above-mentioned environmental example of a publication, the medium type [other] in which data storage is possible and which can be computer read can be used for the above-mentioned example of a system. The medium of other types of these which can be used for the above-mentioned example of operating environment which can be computer read has for example, a magnetic cassette, flash memory card, a digital videodisc, a BERUNUI (Bernoulli) cartridge, random access memory (RAM), the memory (ROM) only for playbacks, etc.
[0063] Many program modules are stored in a hard disk, a magnetic disk 516, an optical disk 519, and ROM508 or RAM510, and these contain the application program 528, other program modules 530, and the program data 532 of 526 or 1 or more operating systems. A user can input a command and information into a computer system 500 with input units, such as a keyboard 534 and a mouse 536, or other pointing equipments. As other input devices, there are a microphone, a joystick, a gamepad, a satellite dish, a scanner, etc., for example. These and other input devices are connected to the processor 502 through the serial port interface 540 connected to the system bus 506 in many cases. However, these input devices may be connected by other interfaces, such as a parallel port, a game port, or Universal Serial Bus (USB), again. The monitor 542 or the indicating equipment of other types is also connected with the system bus 506 through the interface of a video adapter 544 etc. In addition to a monitor 542, typically, a computer system contains other circumference output units (not shown), such as a loudspeaker and a printer.
[0064] A computer system 500 can operate in the environment using the logical connection to one or more remote computers like a remote computer 546 connected by network. a remote computer 546 -- a computer system, a server, a router, Network PC, and a pier (peer) -- it may be equipment or other common network nodes, and there are many elements typically mentioned above in connection with a computer system 500, or all are included. Network connection contains Local Area Network (LAN) 548 and Wide Area Network (WAN) 550. Such a network environment is not new in office, an enterprise

magnitude computer network, intranet, and the Internet.
[0065] When using by the LAN network environment, a computer system 500 is connected to a local network 548 through a network interface or an adapter 552. When using by the WAN network environment, a computer system 500 includes the modem 554 or other means for establishing the communication link by Wide Area Network 550 like the Internet typically. Built-in or external any is sufficient as a modem 554, and it is connected with the system bus 506 through the serial port interface 540. In the environment connected by network, the program module described in relation to the computer system 500 or its part may be memorized by remote memory storage. The illustrated network connection is an example and can use other means for the communication link establishment between computers.
[0066] In the operation gestalt of this invention, a computer 500 expresses a web server and CPU502 carries out executive operation of the page factory module on the ASP+ resource memorized by at least one of storages 516, 512, 514, 518, and 519 or memory 504. A HTTP response and a request communicate by LAN548 connected to the client computer 546.
[0067] Drawing 6 is a process flowchart showing server side processing of the page object in an operation gestalt with this invention, and other control objects. In processing 600, the page object construction section is called with the page factory module 308 (refer to drawing 3). Consequently, a page object (see the page object 314 in drawing 3) is created so that it may correspond to the web page user interface element and logic target on a client. In processing 602, a page factory module calls the ProcessRequest() member function of the page object which begins to process [gradual] the HTTP request received from the client. In the 1st phase of 1 operation gestalt of this invention, server side creation processing (not shown) is creation of the descendant server side control object contained in the control object hierarchy of a page object. That is, in order that the construction section of a child control object may create the control object between the processing life times of HTTP request processing, call appearance is carried out repeatedly.
[0068] However, in another operation gestalt, creation of a child control object is delayable until a control object is needed for a given processing step (for example, rendering of the HTML code of the user interface element with which handling of a postback event, handling of postback data, loading of a view state and preservation, and data coupling decompose or correspond). Since the operation gestalt of the latter "which delays control object creation" can reduce utilization of unnecessary CPU and memory, it is the optimal. For example, there is a case where it will be called creation of the web page from which the user input event received from the client completely differs. In this case, a control object hierarchy makes it end promptly, and only in order to process the event which will illustrate a new different control object hierarchy of a new page, it is not necessary to illustrate the whole control object hierarchy of a former page.
[0069] The server call of the ProcessRequest method of a page object can be answered, and executive operation of the processings 604-620 can be carried out by the data page object and each descendant control object according to the data of a given HTTP request etc. In an operation gestalt with this invention, processings 604-620 are performed to each object of each in the sequence of drawing 6 . However, depending on a HTTP request, about given processing of another object, it may not be carried out in order or, as for the given processing to one object, processing may not be performed at all. For example, the 1st object performs the initialization processing 604 and its load processing 606, the postback data processing 608 is started and a descendant control object performs the initialization processing 604 and the load processing 606 of itself by overdue control object creation after that. Although the sequence of processing by the page object and the descendant control object is not restricted to this, it depends on the various factors [whether the current condition of the property of the data in a HTTP request, a control object hierarchy's configuration, and a control object and control object creation are performed behind time] to include.
[0070] The initialization processing 604 initializes a control object, after a control object is created by carrying out executive operation of the server side code of the arbitration about initialization in a dynamic contents resource. Thus, each server side control object may be customized by the specific

server side function declared with the dynamic contents resource. In the operation gestalt of this invention, a dynamic contents code customizes or extends the base page control class declared by the page developer with the ASP+ resource on a server. Compile of an ASP+ resource contains the declared code in the suitable initial code (for example, initialization () method of a page object and a descendant control object). The initialization processing 604 carries out executive operation of this code, and customizes or extends a page base class and the base class of a descendant control object.
[0071] In an operation gestalt with this invention, the status management of a server side control object is supported in the load processing 606 and the preservation processing 616 in which the condition structure which can be conveyed is used, in order to contain the non-statement model of a client/server system by returning a server side control object to a former condition. With a certain operation gestalt, although a condition goes and comes back to a server in the one or more HTML fields in which the HTTP request / response of a couple hid, it is considered that other condition structures which can be conveyed are within the limits of this invention.
[0072] In a series of processings by the given present request and present given response about a page between a client and a server, the condition of one or more control objects is recorded on the condition structure which can be conveyed by the preservation processing 616 after processing of a previous request. In an operation gestalt with this invention, the status information of the addition containing the control object identifier which enables a hierarchy or a server to associate a suitable control object and a given condition is also included in the condition structure which can be conveyed. In the next HTTP request, status information is returned to a server with the condition structure which can be conveyed. A server extracts status information from the received condition structure which can be conveyed, loads condition data to the suitable control object in a control object hierarchy, and returns each control object to the condition that it existed before the previous HTTP response. After the processing to the present request, again, the condition of one or more server side control objects is recorded on the condition structure which can be conveyed by the preservation processing 616, and returns the condition structure which can be conveyed to a client by the next HTTP response by it.
[0073] Each server side control object is changed into the same condition as the condition before a previous HTTP request as a result of the load processing 606. For example, when a text box control object includes a property value equivalent to "JDoe" before a previous HTTP response, the load processing 606 returns the same control object to a previous condition by loading text string "JDoe" to the property value etc. Furthermore, whether the condition of a given object was memorized and recovered can also constitute.
[0074] When 1 operation gestalt of this invention is summarized, "it is saved" after the condition of one or more server side control objects processing. The saved status information is transmitted to a client by the response. A client returns the status information saved by the next response to a server side. A server loads status information to the newly illustrated server side control object hierarchy so that a hierarchy's condition may return to a former condition.
[0075] With another operation gestalt, status information can be held for some other accessible web locations by the server during round trip on a server of returning from a server to a client and a server. After a server receives a client request, this status information is taken out by the server and may be loaded to the suitable server side control object in a control object hierarchy.
[0076] In processing 608, the postback data received from the HTTP request are processed. Postback data may be contained in the pay load of the HTTP request by the key value and hierarchical display (for example, XML) which became a pair, or other data display like RDF (Resource Description Framework). Processing 608 analyzes the syntax of a pay load, and identifies the identifier of the meaning of a server side control object. When an identifier (for example, "page1:text1") is found and the identified server side control object exists in a control object hierarchy, corresponding postback data are passed to the control object. For example, reference of drawing 1 sends the identifier of a meaning relevant to a text box 106 and text "JDoe" to a web server 116 with the pay load of the HTTP request 114. Processing 608 analyzes the syntax of the pay load of the HTTP request 114, and acquires the identifier and its related value (namely, "JDoe") of a meaning of a text box 106. And processing 608

decomposes the identifier of the meaning of a text box 106, identifies a corresponding server side control object, and passes a "JDoe" value to the object to process.
[0077] As the load processing 606 was explained, the property value of a server side control object may be returned to a former condition. If postback data are received, a server side control object will judge whether the property value of the point to which the passed postback value corresponds was changed. When it changes, logging of the change is carried out to the change list in which data change of a related control object is shown. After all postback data are processed within a control object hierarchy, a control object method is called and one or more postdata change events can be raised to one or more non-user interface server components like stock price retrieval application started on a server. A postdata change event is an event which shows that for example, postback data changed the property of a server side control object. In an instantiation operation gestalt, such an event is sent to a system offer event queue, and can call the server side code registered so that an event might be processed. And a server side code can call the method of a non-user interface server component. Thus, a server side non-user interface server component can answer the event in which the trigger was carried out by change of the data of a server side control object. The option including using an application offer event queue, polling, and processing interruption which performs an event can also be considered within the limits of this invention.
[0078] A postback event is handled in processing 610. A postback event can communicate with the pay load of a HTTP request. Processing 610 analyzes the syntax of the specific event target (for example, with the operation gestalt with this invention, label attachment is carried out with "##EVENTTARGET") which identifies the server side control object to which the event is turned. Furthermore, processing 610 analyzes the syntax of it, when there is a discovered event argument, and it gives it to the server side control object which had the event argument (for example, label attachment is carried out with "##EVENTARGUMENT" with the operation gestalt with this invention) specified. A control object starts the event processed in server side code which calls the method of the non-user interface server component (for example, server side stock price retrieval application) relevant to a dynamic contents resource.
[0079] Processing 612 solves the data-coupling relation between a server side control object and one or more databases with an accessible server, and updates it in a control object property by this with a database value, and updates the database field with the value of/or a control object property. the property of a server side control object may be related with the property of a parent data-coupling container as shown in the table of a server side application database with an operation gestalt with this invention (or data coupling is carried out -- having). A page frame work piece can update the control object property which has the value of the parent data-coupling container property corresponding to between the data-coupling processings 612 and by which data coupling was carried out. Thus, the user interface element on the web page in the next response reflects the updated property value in accuracy. The control object property with which a user interface element corresponds is because it was automatically updated between the data-coupling processings 612. Similarly, a control object property may be updated by the parent data-coupling container field, and, thereby, updates a server side application database in the postback input from a server side control object again.
[0080] Processing 614 updates a large number by which executive operation may be carried out, before saving a control object condition and carrying out the rendering of the output. Processing 616 requires status information (namely, view state) from one or more control objects in a control object hierarchy, stores status information, and inserts it in the condition structure which is sent to a client with a HTTP response pay load and which can be conveyed. For example, a "grid" control object can save the present index page of the list of values, and a "grid" control object can return to this condition after the next HTTP request (namely, processing 606). As mentioned above, view state information expresses the condition of the control object hierarchy before the next action by the client. It will be used for changing a control object hierarchy into the condition of the point before client post back input process or data coupling if view state information returns.
[0081] The rendering processing 618 generates the suitable authoring language output (for example,

HTML data) sent to a client by the HTTP response. A rendering is performed by top-down hierarchy tree WOKU and the embedded rendering code of all server side control objects. Processing 620 does the last clean-up activity (it is connection of a database in closing a file ****) of arbitration, and a control object hierarchy is terminated. Subsequently, processing performs return and processing 622 to 602, and a page object is ended by calling the destructive section there.
[0082] Drawing 7 expresses the notation of an example of the server side control class in an operation gestalt with this invention. The server side control class defines the method common to all the server side control objects in an operation gestalt with this invention, the property, and the event. The more concrete control class (for example, server side carbon button control object corresponding to the client side carbon button in a web page) is derived from this control class. The shown control class 700 contains the memory which stores a property 702 and a method 704. The operation gestalt of other control classes from which the combination of a data member and a method differs is also considered within the limits of this invention.
[0083] In the shown operation gestalt, the property 702 is public. Property"ID" is a string value in which reading which shows the control object identifier and writing are possible. Property "Visible" is a boolean value in which reading which shows whether the rendering of the authoring language data of a corresponding client side user interface element should be carried out and writing are possible. Property "MaintainState" is a boolean value in which reading which shows whether a control object should save the view state (and view state of the child object) at the time of termination of the present page request (by namely, response to the preservation processing 616 of drawing 6) and writing are possible. Property "Parent" is a reference in which reading by the control container (refer to drawing 8) related in a control object hierarchy's present control object is possible. Property "Page" is a reference in which reading by the root page object to which it acts as the host of the current control object is possible. Property "Form" is a reference in which reading by the form control object to which it acts as the host of the current control object is possible. Property "Trace" is a reference which enables the writing of a developer's trace log and which can be read. A property "BindingContainer" is a reference in which reading by the immediate-data joint container of a control object is possible. Property "Bindings" is a reference in which reading by the set of control object data-coupling association is possible.
[0084] A method 704 includes the access approach to processing of a request, and the data member of a control object. A method is referred in a certain operation gestalt by the pointer stored in the memory tooth space of a control object. This reference means is used in the operation gestalt of ** and a container control object, a control collection object, and other server side objects containing a page object. Method"Init()" is used for initializing these after a child control object is created (see the processing 604 of drawing 6). Method"Load()" is used for recovering view state information from a previous HTTP request (see the processing 606 of drawing 6). Method"Save()" is used for saving the view state information that it uses at a next HTTP request (see the processing 616 of drawing 6). Method"PreRender()" is used for performing a required pre rendering step in advance of preservation of a view state, and the rendering of contents (see the processing 614 of drawing 6). Method"Render (TextWriter output)" is used for outputting the authoring language code of the user interface element corresponding to the present control object (see the processing 618 of drawing 6). Since a code is stored by the response to a client, a code lets an output stream pass and is carried out (it is passed with an "output" parameter). Before method"Dispose()" ends a control object, it is used for doing a final clean-up activity (see the processing 620 of drawing 6).
[0085] Method"GetUniqueID()" obtains the string identifier as which the meaning of a control object was recognized hierarchical. Method"GetControlWithID(String id)" returns the reference to the direct child control object which has the given identifier ("id"). Method"GetControlUniqueWithID(String id)" returns the reference to the child control object which has the hierarchy identifier ("id") of a meaning.
[0086] Method"PushControlPropertyTwoBindingContainer" (String prop Name) will be used for updating the joint container of 2-way data coupling from postback data if a postback data value changes within a server side control object. Method"PushBindingContainerPropertyTwoControl(String prop Name)" is used for updating a server side control object property with a current joint container value.

Method"HasBindings()" returns the boolean value which shows whether the control object has joint association. These three functions are used for solving the joint relation between the property of a server side control object, and the attribute in the server side data storage section (see the data-coupling processing 612 of drawing 6).
[0087] Drawing 8 shows an example of the server side container control class in an operation gestalt with this invention. A container control class supports a nested child control object, and offers serialization and the un-serialized configuration for view state information automatically to the condition structure which can be conveyed. The container control class 800 contains the memory which stores a property 802 and a method 804. Property "Controls" is a reference in which reading by "ControlCollection" of the child control object in a control object hierarchy is possible (see drawing 9). Property "StateCollection" is a reference in which reading by the dictionary of view state information used for maintaining the condition of a control object over much page requests is possible. Method"HasControls()" returns the boolean value which shows whether the control object has the child control object.
[0088] In another operation gestalt, the member property and method of the container control class 800 may be included in the control class 700 of drawing 7 so that each control object can support a child object in itself. However, if such a control object of an operation gestalt does not contain such a child object, a control member (control collection type) will be empty (that is, a child object is not included).
[0089] Drawing 9 shows an example of the server side control collection class in the operation gestalt of this invention. The control collection class 900 contains the memory which stores a property 902 and a method 904. Property "All" is a snap shot dump array in which reading and the writing of all the child control objects of the current control object ordered by the index are possible. Property"this[index]" is a reference in which reading by the control object of the sequence index in a control collection is possible. Property "Count" is a value which shows the number of the child control objects in a collection and which can be read.
[0090] Method"Add'Control child" is used for adding a specific control object to a current collection. Method"IndexOf(Control child)" returns the sequence index of a child control object with which it was specified in the collection. Method"GetEnumerator(boolAllowRemove)" returns ENYUMERETA of all the child control objects in a collection. Method"Remove(Control value)" removes a specific control object from a current collection. Method"Remove(int index)" removes a specific control object from a current collection based on the given index. Method"Clear()" removes all control objects from a current collection.
[0091] Drawing 10 shows an example of the server side page object in an operation gestalt with this invention. A page base class defines the method common to all server side actuation pages, property, and event within the operation gestalt of this invention. The page object 1000 contains the memory which stores the property 1002 and the method 1004. Property "ErrorPage" is a string in whom reading by which a rendering is carried out in the event of an unsettled page exception and writing are possible. Thus, a page object returns the error page which can be read to a client by the HTTP response. Property "Requests" is a reference in which reading by HTTPRequest is possible, and HTTPRequest is given according to the web server framework which gives functionality for close to access the HTTP request data to which it comes.
[0092] Property "Response" is a reference in which reading by the HTTP response given according to the web server framework which offers the function for transmitting HTTP response data to a client is possible. Property "Application" is a reference in which reading by HTTPApplication given according to a web server framework is possible. Property "Session" is a reference in which reading by HTTPSession given according to a web server framework is possible. Property "Server" is a reference in which reading by the server object which is an application server page compatibility utility object is possible. Property "Context" is a reference in which reading of all the web server objects given according to a web server framework is possible, and, thereby, as for a developer, access to the further pipeline module exposure object is attained. property "IsFirstLoad" -- a page object -- for the first time -- or it is the boolean value which shows whether it is loaded and accessed by the response to a client post back

request and which can be read.
[0093] Property"Load()" initializes a page object and is used for recovering view state information from a previous page request. Property"Save()" is used for saving the view state information that it uses for a next page request. Property"HandleError(Exception errorInfo)" is used for treating the unsettled error which takes place between page executive operation. In this event, base class activation returns a client to URL which has a default error web page. Method"GetPostbackScript(Control target, String name, String arg)" returns the client side script method relevant to a given control object. Method"RegisterClientScriptBlock(String key, String script)" is used for eliminating the duplication block of the client side script code transmitted to a client. A duplication block is a script which has the same key value. Method"IHTTPHandler.ProcessRequest(HTTPContext Context)" is used for processing a web request. Since processing of the HTTP request received from the client is initialized, IHTTPHandler.ProcessRequest is called (see the processing 602 of drawing 6). Method"IHTTPHandler.IsReusable()" shows whether a page object can reuse, although much web requests are processed.
[0094] The operation gestalt of this invention given in this description is performed as a logic step of one or more computer systems. Logic processing of this invention is performed as a machine module with which it interconnected within the computer system beyond (2) 1 ** as a processor execute step sequence by which executive operation is carried out by the computer system beyond (1) 1 **. Activation is the problem of selection and it depends for it on the performance requirements of the computer system which performs this invention. Therefore, the logic processing which constitutes the operation gestalt of this invention of a publication on these descriptions can be variously expressed as processing, a step, an object, or a module.
[0095] A description, an above-mentioned example, and above-mentioned data offer the structure of the operation gestalt of this invention, and perfect explanation of an activity. Since this invention can be carried out with many gestalten, without deviating from the pneuma and the range of this invention, this invention is in an attached claim.
[0096]
[Effect of the Invention]

[Translation done.]

TECHNICAL FIELD

[Field of the Invention] Generally especially this invention relates to the server side control object which processes the client side user interface element of a web page about a web server framework.

[Translation done.]

PRIOR ART

[Description of the Prior Art] A typical web browser receives data from the web server which defines the appearance of a web page and basic actuation which are displayed in a client system. As a typical procedure, a user specifies the uniform resource (resource) locator (henceforth "URL") which is the global address of the resource on World Wide Web, and a desired website is accessed. Generally, the vocabulary "a resource" is the data which can be accessed by the program, or a routine.
[0003] An example of URL is "HYPERLINK "http://www.microsoft.com/ms.htm" http://www.microsoft.com/ms.htm." The 1st part of this example of URL shows the given protocol (for example, "http") used for a communication link. The 2nd part specifies the domain name (for example, "HYPERLINK "http://www.microsoft.com" www.microsoft.com") which shows the location of a resource. The 3rd part specifies the resource in a domain (for example, file called "ms.htm"). This is followed and they are a browser and HYPERLINK"http://www.microsoft.com" www.microsoft.com. The HTTP (hypertext transport protocol) request relevant to the example of URL for taking out the data relevant to the ms.htm file in a domain is generated. The web server which is acting as the host of the www.microsoft.com site receives a HTTP request, returns the demanded web page or resource to a client system by the HTTP response, and displays it on a browser.
[0004] The "ms.htm" file of the above-mentioned example contains the static HTML (HyperText Markup Language) code. HTML is a plane (plaintext) text authoring language used for the document (for example, web page) creation on World Wide Web. Since it is such, an HTML file can be displayed on a browser as a web page, in order to offer graphical experience which a user expects, while being taken out from a web server and displaying the information from the Internet.
[0005] A developer can specify the text and list which are displayed on a browser and by which formatting was carried out, form, a table, a hypertext link, an in-line image, voice, and background graphics using HTML. However, an HTML file is a static file which is not supporting dynamic generation of web page contents in essence.
[0006] Moreover, a web page may need to display dynamic contents, such as change, traffic information, etc. on a stock price, on a browser. In such a situation, a typical server side application program acquires dynamic data, and it is developed so that it may be formatted into the HTML format transmitted to the browser displayed on a web page while a web page is updated.
[0007] Furthermore, although data are not strictly dynamic, since the value displayed on a static web page is a value from which many differ dramatically, in a situation it is not practical to create a demanded number of static web pages, these same servers side application program can be used. For example, a travel plan page may give two kinds of calender indication with the calender for start days, and the calender for return days. Instead of developing the static page of hundreds by the combination of all the calenders that can be considered, a server side application program can generate dynamically the suitable static page which displayed the suitable calender.
[0008] By many web pages, a user can have a dialog with the page displayed on the browser by choosing the visual element of a page. For example, in the above-mentioned travel plan page, with a calender, or a user clicks the date and chooses the date, he can have a dialog with the calender by

clicking on an icon, carrying forward the moon or returning. A browser advances a HTTP request to a server side application program by the existing solution. This HTTP request can contain the parameter coded in addition to this in an enquiry matrix like a form post variable, or the data format which describes a client side event or data (for example, which control did the user click?). For example, a parameter may contain the data which the user chose in one calender with data current on display in the calender of another side.
[0009] The communication link of the event and data which are returned to a server is called the "postback." This is because a browser transmits a request using a HTTP post request typically. A server side application program processes a HTTP request, and generates the suitable HTML code for the web page which has the newly calculated calender reflecting action of the user who transmits to a client by the HTTP response. Then, the obtained document is transmitted to a client system by the HTTP response, and this document is displayed on a browser as a web page which shows the updated calender.

[0010] That it is well versed in the pro GURAMUBE six which data are transmitted between a browser and a server how development of a server side application program is not only well versed in the usual HTML coding used for a web page design, but, or contains one or more programming language (for example, C++, Perl, Visual Basic, or Jscript) and HTTP protocols is the complicated activity demanded. However, a web page architect is graphic designer or an editor in many cases, and may be inexperienced in a program. Furthermore, if complicated web page development is simplified, the rate of development of new web contents can be gathered by any developers.
[0011] Generally, great efforts are required and development of a custom-made server side application program is also actually like [which a developer regards as not wanting to often try it]. In order to display a desired web page, a developer not only understands the HTML code as which generation is required, but has to understand what the user dialogue and client data from a web page become postback processing. Therefore, it is desirable to offer the development framework to which a developer can create and process a web page dynamically by the minimum programming.

[Translation done.]

EFFECT OF THE INVENTION

[Effect of the Invention]

[Translation done.]

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] One means which makes min the requirements for a program of dynamic web page generation is an active server page (ASP) framework offered by Microsoft Corp. An ASP resource processes typically the HTTP request which specifies an ASP resource as a desired resource, after that, generates the HTML code as a result of the HTTP response to a client, for example, contains Visual Basic or Jscript. Furthermore, in order to ease given application programming efforts, or the ASP resource was developed beforehand, refer to a third party's client side library component (for example, client side "ACTIVEX" control) for it. However, in the present server side application framework, programming which must manage dynamically client side user interface elements (for example, a text box, a list box, a carbon button, a hyperlink, an image, voice, etc.) within server side application needs a still advanced programming technique and considerable efforts. An unsolved technical problem is about things for which a user interface element is processed, such as handling a postback event, encapsulating programming demanded appropriately so that a web page developer can focus on other aspects of a web page.

[Translation done.]

MEANS

[Means for Solving the Problem] According to this invention, **** and other technical problems are solved by offering the server side control object framework which manages processing and generation of a client side user interface element. Furthermore, the hierarchy of a server side control object can collaborate and generate an authoring language code a result like the criterion HTML for displaying a web page on a client. As long as a client supports an authoring language code as a result of [another] for example, the criterion HTML, it may be what kind of browser. Processing of a client side user interface element includes one or more postback event handling processings, postback data handling processing, data-coupling processing, or the status management processing about the condition of a server side control object.
[0014] The big effectiveness of an operation gestalt with this invention is in improvement in capsulation of server side processing of a client side user interface element (for example, output used for the reception input from a user interface element, and generation of a user interface element), and related functionality. One or more server side control objects may be generated so that it may correspond to one or more user interface elements logically. For example, considering the user interface element which expresses the moon in a KARENTA display, the hierarchy of a server side control object may be generated corresponding to a calender display and its various sub element. In the "moon" control object, with a certain configuration, the ** "week" control object contains seven "day" control objects hierarchical, including many "week" control objects hierarchical.
[0015] Furthermore, in an operation gestalt with this invention, a server side control object can collaborate and process the user interface element which corresponds logically. This advantage is the result of being brought when many server side control objects exist simultaneously during client request processing and generation of a response. For example, if the 1st postback event (for example, click of the carbon button element of the "next moon" on a calender display) which received from the client is detected, one control object will start the 2nd event (for example, the selection and the display of the next moon by the "moon" control object) which is detected after that and processed by one or more control objects which are recognizing simultaneous existence. By this collaboration, a complicated control dialogue can be encapsulated within the server side control object itself, and, thereby, the custom-made event handling required of a web page developer can be made into the minimum.
[0016] The approach and computer program product which process the client side user interface element built into the web page displayed on a client are offered. The request which refers to the server side resource data storage section is received. Declaration is inputted from the server side declaration data storage section. A server side control object is generated and programmed to give the functionality of a user interface element based on declaration. A user interface element is processed using a server side control object. Authoring language data are generated from the server side control object which displays a user interface element on a web page.
[0017] The hierarchy of the server side control object in which executive operation is possible is offered by computer which processes one or more client side user interface elements built into the web page displayed on a client. One or more server side child objects correspond to one or more client side user

interface elements. Each server side child object treats the input received from the client, and generates the authoring language data for displaying a client side user interface element on a client. At least one hierarchy identifier is received from the client relevant to the input which specifies one of server side child objects. A server side page object receives the input distributed to one of the server side child objects according to a hierarchy identifier including a server side child object.
[0018]
[Embodiment of the Invention] The operation gestalt with this invention contains the server side control object which processes and generates the client side user interface element displayed on a web page. Furthermore, the hierarchy of a server side control object can collaborate and generate an authoring language code a result like the criterion HTML for displaying a web page on a client. As long as a client supports an authoring language code as a result of [another] for example, the criterion HTML, it may be what kind of browser. With an operation gestalt with this invention, a server side control object corresponds to a client side user interface element and a logic target, and generates the authoring language code used by the client side browser which displays and processes a web page by the server. Processing of a client side user interface element may also include one or more postback event handling processings, postback data handling processing, data-coupling processing, and status management processing.
[0019] Drawing 1 shows the web server which generates dynamically the web page contents displayed on the client in an operation gestalt with this invention. A client 100 performs the browser 102 which displays the web page 104 on the indicating equipment of a client 100. A client 100 may also include the client computer system which has a display like a video monitor. The "INTERNET EXPLORER" browser currently sold by Microsoft Corp. is an example of the browser 102 in an operation gestalt with this invention. As an example of other browsers, it is "NETSCAPE NAVIGATOR". It reaches. Although there is "MOSAIC" etc., it does not restrict to this. The text box control 106 and two carbon button control 108 and 110 are included in the illustrated web page 104. A browser 102 can receive the HTML code from a web server 116 by the HTTP response 112, and displays the web page described by the HTML code. Although HTML is explained with reference to a certain operation gestalt Not the thing restricted especially but SGML (Standard Generalized Markup Language), XML (eXtensible Markup Language) -- and It is the markup language of the XML base. It is thought that other authoring languages containing WML (Wireless Markup Language) designed so that the content and user interfaces of narrow-band radio equipment, such as a pocket bell (trademark) and a cellular phone, might be specified are within the limits of this invention. Furthermore, although the criterion HTML 3.2 is mainly indicated in this description, any versions of HTML may be contained within the limits of this invention.
[0020] The communication link with a client 100 and a web server 116 can be performed using a series of processings of the HTTP request 114 and the HTTP response 112. Although HTTP is explained with reference to a certain operation gestalt, it is not restricted especially and it is thought that other transport protocols including S-HTTP are within the limits of this invention. In a web server 116, the HTTP pipeline module 118 receives the HTTP request 114, analyzes URL, and calls the suitable handler which processes a request. The web server 116 is equipped with two or more handlers 120 which handle the resource of a different type in the operation gestalt with this invention.
[0021] For example, when URL specifies a static contents resource 122 like an HTML file, a handler 120 accesses the static contents resource 122, and sends the static contents resource 122 to a client 100 by the HTTP response 112 through the HTTP pipeline 118. Or in an operation gestalt with this invention, when URL specifies a dynamic contents resource 124 like an ASP+ resource, a handler 120 accesses the dynamic contents resource 124, processes the contents of the dynamic contents resource 124, and generates the HTML code the result for web page 104. In an operation gestalt with this invention, the result HTML code contains standard HTML3.2 code. Generally, a dynamic contents resource is the server side declaration data storage section (for example, ASP+ resource) which can use the authoring language which describes the web page which should be displayed on a client for generating dynamically. And the HTML code for web pages passes the HTTP pipeline 118, and is sent

to a client 100 by the HTTP response 112.
[0022] During this processing, the handler 120 was beforehand developed again, in order to simplify a development effort, or it can access the library of the 3rd person code. One of such the libraries is the server side class control library 126, and a handler 120 can illustrate from here the server side control object which generates HTML data as a result of processing a user interface element and displaying on a web page. in an operation gestalt with this invention, one or more server side control objects are visible on the web page described by the dynamic contents resource 124 -- it hides-like and maps to one or more user interface elements.
[0023] On the other hand, the 2nd library is a client side control class library 128 like the library containing the "ACTIVEX" component from Microsoft Corp. "ACTIVEX" control is a COM (component object model) object which follows a fixed criterion in the method of a dialogue with a client and other components. Client side "ACTIVEX" control is the component of the COM base as for which downloads to a client automatically and executive operation may be carried out to it by the web browser of a client. A server side ACTIVEX component (not shown) is a component of the COM base which may be performed on a server, in order to achieve the various server side functions in which the server side functionality of stock price retrieval application or a database component is offered. ACTIVEX is indicated by the detail by "understanding of ACTIVEX and OLE" (David Chappelle, the Microsoft press, 1996).
[0024] In contrast with "ACTIVEX" control, the server side control object of the operation gestalt of this invention specified as the dynamic contents resource 124 corresponds to the user interface element with which it is displayed on a client logically. A server side control object can generate the effective HTML code which may contain the locator which refers to for example, a HTML tag and given client side "ACTIVEX" control again. When the browser has already had the code for client side "ACTIVEX" control in the storing system, executive operation of the "ACTIVEX" control is carried out within the web page on a client. If it becomes right [that], a browser will download the code for "ACTIVEX" control from the resource specified by the locator, and will carry out executive operation of the "ACTIVEX" control within the web page on a client. The server side control object in the operation gestalt of this invention can start an event again to the server side "ACTIVEX" object used for performing stock price retrieval application on a server.
[0025] A handler 120 accesses one or more non-user interface server components 130 which carry out executive operation on a web server 116 or another accessible web server again. A non-user interface server component 130 like stock price retrieval application or a database component is referred to in the dynamic contents resource 124 processed by the handler 120, or is related with it. The server side event started by the control object declared with the dynamic contents resource 124 may be processed in server side code which calls the suitable method of the non-user interface server component 130. Consequently, the processing offered by the server side control object can simplify programming of non-user interface server component stereo-NENTO 130 by encapsulating processing and generation of a web page of a user interface element, and, thereby, the developer of the non-user interface server component 130 can be concentrated on development of the function of the application proper instead of a user interface problem.
[0026] Drawing 2 shows the flow chart of processing of the client side user interface element using the server side control object in an operation gestalt with this invention, and generation processing. In processing 200, a client transmits a HTTP request to a server. A HTTP request contains URL which specifies resources, such as an ASP+ resource. In processing 202, a server calls the suitable handler which receives a HTTP request and processes the specified resource. Reading appearance of the ASP+ resource is carried out in processing 203. Processing 204 generates a server side control object hierarchy based on the content of the specified dynamic contents resource (for example, ASP+ resource).
[0027] In processing 206, a control object hierarchy's server side control object performs postback event handling, postback data handling, status management, and one or more processings in data coupling. From a client, the postback event and data (collecting "postback input") from a user interface element are sent to a server, and are processed. Especially a postback event is not restricted and may also contain

the "data change" event from the client side text box element sent to "a mouse click" or a server from a client side carbon button element. Especially postback data may also contain the text in which **** was inputted into the index of the item chosen from for example, the text box element or the drop down box by the user in what is restricted. However, postback processing may be performed by other events and is only performed by the dialogue with a user.
[0028] In processing 208, in order that each server side control object in a hierarchy may generate authoring language data like the HTML code for the display by the web page of a client side user interface element (or rendering), it is called. Although the vocabulary "a rendering" may mean the processing which displays graphics on a user interface, in this description, the vocabulary "a rendering" also means generation processing of the authoring language data which may be interpreted by client application like the browser for a display and a client side function. More detailed explanation of processing 206 and the rendering processing 208 is given in connection with drawing 6 . In a certain operation gestalt, the call of the render() method in each control object is performed using a tree traversal sequence. That is, the call of the render() method of a page object becomes the repetitive traverse covering the suitable server side control object in a hierarchy. As an option which calls the render() method of a suitable control object, approaches, such as event signaling or the object registration technique, may be used. A parenthesis specifies the "render()" level which shows the method of comparing with a data value.
[0029] In an operation gestalt with this invention, actual creation of each server side control object may delay until a server side control object is accessed in processings 206 or 208 (handling of a postback input, loading of a condition, rendering of a control object to the HTML code, etc.). When a server side control object is not accessed for a given request, server processing is optimized by delaying creation of a control object and eliminating unnecessary control object creation processing.
[0030] In processing 210, the HTML code is transmitted to a client by the HTTP response. In processing 214, a client receives the HTML code relevant to the new web page which should be displayed. In processing 216, a client system incorporates the user interface element of a new page according to the HTML code received from the HTTP response (for example, it displays). however, a user interface element -- incorporating -- he should also understand that you may include non-display processing of control of read-out to the memory which offers voice or a tactile-sense output and writing, and script processing etc. A server side control object hierarchy is ended in processing 212. In an operation gestalt with this invention, the server side control object in a hierarchy answers the HTTP request which refers to the ASP+ response of relation, is created, and after the rendering of authoring language data (for example, HTML data) finishes, it is destroyed. In another operation gestalt, processing 212 may be performed before processing 210 after processing 208.
[0031] Drawing 3 shows an example of the module in the web server used in an operation gestalt with this invention. A web server 300 accepts the HTTP request 302 in the HTTP pipeline 304. The HTTP pipeline 304 may also contain various modules, such as logging of web page statistics, user collating, a right of user access, and a module for the formation of an output cache of a web page. Eventually, each input HTTP request 302 received by the web server 300 is processed by the specific instance of an IHTTP handler class (it illustrates as a handler 306). A handler 306 decomposes a URL request and calls a suitable handler factory (for example, page factory module 308).
[0032] In drawing 3 , the page factory 308 related with the ASP+ resource 310 is called, and instantiation and configuration of the ASP+ resource 310 are handled. In a certain operation gestalt, an ASP+ resource may be recognized by specifying a specific suffix (or a file extension part like ".aspx") as a file. If the request to given ".aspx" resource is first received by the page factory module 308, the page factory module 308 will search a file system, and will obtain a suitable file (for example, .aspx file 310). This file may also contain the text (for example, authoring language data) which may be behind interpreted or accessed by the server which processes a request, or the data (for example, cutting tool code data or coded data) in another format. When a physical file exists, the page factory module 308 reads a file to an aperture, and reads the file to memory. When a file is not found, the page factory module 308 returns the suitable error message "a file is not found."

[0033] After reading the ASP+ resource 310 to memory, the page factory module 308 processes a file content, and builds the data models (for example, a list [of script blocks], directive, static text area, and hierarchy server side control object, a server side control property, etc.) of a page. A data model is used for generating the source list of a new object class like the COM+ (Component Object Model+) class to which the class of the page base is made to extend. The page base class contains the code which specifies the structure, property, and function of a basic page object. A source list is dynamically compiled by intermediate language and is behind compiled [in / next / an operation gestalt with this invention] by the instructions (for example, X86, Alpha, etc.) of a proper on a plat form timely (Just-In-Time). Intermediate language is COM+ IL. A code and Java A cutting tool code and Modula 3 A code and SmallTalk A general purpose or custom-made orientation linguistic code of a code, the Visual Basic code, etc. may also be included. In another operation gestalt, intermediate-language processing may be excluded and a native instruction may be directly generated from a source list or a source file (for example, ASP+ resource 310). The control class library 312 may be accessed with the page factory module 308, in order to obtain the server side control class which is used for a control object hierarchy's generation and which was defined beforehand.
[0034] The page factory module 308 outputs the page object 314 which is a server side control object equivalent to the web page 104 of drawing 1 . The page object 314 and its child object (for example, the text box object 318, the carbon button object 320, and another carbon button object 322) are the control object hierarchy's 316 examples. The example of other control objects can be considered according to this invention, like the custom-made control object, is not restricted especially and also contains the object corresponding to HTML control of a table 1. The page object 314 corresponds to the web page 104 of drawing 1 . The text box object 318 corresponds to the text box 106 of drawing 1 . Similarly, the carbon button object 320 corresponds to the additional carbon button 108 of drawing 1 , and the carbon button object 322 corresponds to the deletion carbon button 110 of drawing 1 . The page object 314 relates to other control objects and hierarchy targets on a server. In a certain operation gestalt, a page object is a container object which contains the child control object hierarchical. The hierarchical relationship of other gestalten, such as a dependency, can be used with another operation gestalt. In a complicated control object hierarchy, one child object may be a container object of other child objects rather than it has the child object of a large number level.
[0035] It sets in the above-mentioned operation gestalt, the control object hierarchy's 316 control object is created and performed on a server 300, and each server side control object corresponds to the corresponding user interface element and corresponding logic target on a client. A server side control object collaborates, handles the postback input from the HTTP request 302 again, manages the condition of a server side control object, performs data coupling with a server side database, and generates the authoring language data (for example, the HTML code) used for the display of a web page as a result of a client. Result authoring language data are generated from the server side control object hierarchy 316 (namely, rendering), and are transmitted to a client by the HTTP response 324. For example, the result HTML code can materialize any effective HTML configurations, and in addition to this, control of an ACTIVEX (trademark) type, a JAVA (trademark) applet, a script, and when it is processed by the browser, refer to any web resources which produce client side user interface elements (for example, a control carbon button, a text box, etc.) for it.
[0036] By declaration made with the ASP+ resource 310, a server side control object can access one or more non-user interface server components 330 for a dialogue with the non-user interface server component 330 and a client side user interface element. For example, a postback input can be answered and a server side control object can start a server side event to the non-user interface server component registered into those events. Thus, the non-user interface server component 330 minds a user interface element for a dialogue with a user, and it can perform it, without programming a code required displaying and processing these elements.
[0037] Drawing 4 shows the content of an example of the dynamic contents resource in an operation gestalt with this invention. In the illustrated example, the file 400 includes plain text declaration in a certain dynamic contents resource format (for example, ASP+). Each declaration gives an instruction to

the page compiler which reads a file 400, in order to process a client side user interface element, it performs creation and a call of a suitable server side control object, and it combines the HTML code which is eventually transmitted to a client by the HTTP response and by which the rendering was carried out. And describe or refer to the function of the client side user interface element performed by the server side control object for declaration. And a server side control object generates the HTML code used in order to define the new version of the web page on a client.
[0038] The 1st line of a file 400 is :<%@ directive containing the directive of the following formats. [attribute=value] %> It is here, and especially directive is not restricted and may also contain "page", "cache", or "import". In order to determine buffering semantics, the requirements for a session condition, an error handling scheme, scree PUTINGU language, transaction semantics, and a property like an import directive, a directive is used by the page compiler, when processing a dynamic contents resource. A directive may exist anywhere in a page file.
[0039] <html> of the 2nd line is a standard HTML initiation tag passed to the result HTML code as a literal (that is, additional processing is not performed in order to carry out the rendering of the result HTML code). In HTML functor, the beginning of an HTML file is shown and it has become the termination tag </html> of the line 21 this [whose] is also a literal, and a pair.
[0040] A code declaration block exists in the lines 3-10 of a file 400. Generally, a server side code declaration block defines the method by which executive operation is carried out on a page object, a control object member variable, and a server. the following formats -- setting -- :<script runat = "server" [language = "language"] and [src = "externalfile"] -- > .... </script> -- here, language and a src parameter are arbitrary. In an operation gestalt with this invention, a code declaration block is defined using the <script> tag including the "runat" attribute which has the value set as the "server." Since the syntax of an inner code is specified, a "language" attribute may be used for arbitration. Although default language can express the language configuration of a whole page, a developer is Jscript by the "language" attribute of a code declaration block. And different language within the same (Practical Extraction and Report Language) web page activation of PERL etc. can be used. The <script> tag can specify a "src" file as arbitration again, and a "src" file is a file of the exterior where a code is inserted in the dynamic contents resource for processing by the page compiler from there. Although the syntax currently indicated is used in this operation gestalt, with another operation gestalt, he should understand that different syntax within the limits of this invention can be used.
[0041] In drawing 4 , two subroutines, AddButton#Click, and DeleteButton#Click are declared in the Visual Basic format into the code declaration block. If any subroutine takes two input parameters, "Source", and "E" and a client side click event is detected by the carbon button of a response, executive operation will be carried out on a server by the response to a HTTP request. In an AddButton#Click subroutine, the text in a user name text box is connected with word "Add", and is loaded to the text data member of a message. In a DeleteButton#Click subroutine, the text in a user name text box is connected with word "Delete", and is loaded to the text data member of a message. Although not shown in drawing 4 , the member variable of a server side control object may be declared with the code declaration block of a file 400. For example, if Visual Basic syntax is used, keyword tooth-space"DIM" will declare the data variable of a server side control object.
[0042] "A code rendering block" (not shown) may be included in a dynamic contents resource. In an operation gestalt with this invention, executive operation of the code rendering block is carried out by one "rendering" method by which executive operation is carried out at the time of a page rendering. A code rendering block fills the following formats (other formats are considered in another operation gestalt).
[0043] <% InlineCode %> "InlineCode" includes the independent language mold code block or flows-of-control block which carries out executive operation on a server at the time of a rendering here.
[0044] An in-line expression can also be used within a code rendering block using the following instantiation-syntax.
[0045] <%= InlineExpression %> Here, the expression included in the "InlineExpression" block is eventually included by the call to "Response.Write (InlineExpression)" of the page object which writes

the value from "InlineExpression" in the holder of the suitable location in declaration. For example, "InlineExpression" may be contained in the following code rendering blocks.
[0046] <font size = "<%=x%>" > Hi <%=Name%>, you are <%=Age%>! </font> This outputs a greeting and the description about a certain man's age with the font stored in value"x." The man's identifier and age are defined as a string in a code declaration block (not shown). The rendering of the result HTML code is carried out by the server, and it is transmitted to a client by the HTTP response so that the value of "InlineExpression" may be included in a suitable location. That is, it is as follows.
[0047] <font size ="12"> Hi Bob and you are 35! In the line 11 of a file 400, <body> is a standard HTML tag for specifying the beginning of the text of a HTML document. The termination tag </body> is shown in the line 20 of FIIRU 400. In an operation gestalt with this invention, both <body> and </body> are literals.
[0048] The initiation tag <form> of a HTML foam block is seen on a line 12 in this section of the file 400 of drawing 4 . The termination tag </form> of a foam block is seen on the line 19 of HTML file 400. Since a given identifier is related with a foam block, parameter "id" of arbitration can also be included in a HTML control tag, and it enables this to contain many foam blocks in one HTML file.
[0049] The server side label identified by the "message" is declared in the line 18 of a file 400. A "message" label is used in code declared with the lines 5 and 8 of a file 400, in order to display a label on a web page.
[0050] In the foam block, three examples of a HTML control tag corresponding to the user interface elements 106, 108, and 110 of drawing 1 are shown. The 1st user interface element is declared with the line 13 of the file 400 equivalent to a text box. Text literal "User Name" declares the label located in the left-hand side of a text box. The input tag which has type="Text" declares the text box server side control object which considers as the server side control object which carries out the rendering of the text box client side user interface element, and has an identifier called "UserName". The lines 15 and 16 of a file 400 declare the client side user interface element shown as carbon buttons 108 and 110 of drawing 1 , respectively. A "OnServerClick" parameter specifies the suitable subroutine declared with the code declaration block of a file 400. And the server side carbon button control object generated by the response to declaration by the file 400 carries out the rendering of the server side code of the relation which performs the HTML code and carbon button click event of a client side carbon button.
[0051] The text box and carbon button which were declared by the file 400 are the example of HTML server control declaration. In an initial state, the HTML tags in an ASP+ resource are dealt with [no] as literal text contents, and can be accessed in a page developer's programming. However, when a page developer specifies using the "runat" attribute which has the value set as "server" can show that the syntax of a HTML tag should be analyzed and it should be treated as accessible server control declaration. Each server side control object can be related with the "id" attribute of the meaning which enables program reference of a corresponding control object at arbitration. The property argument on a server side control object and event association can also be specified using the declaration name / value which is an attribute pair in a tag element (for example, OnServerClick is equal to a "MyButton#Click" pair).
[0052] In an operation gestalt with this invention, the general syntax which declares a HTML control object is as follows.
[0053]
<HTMLTag id = "Optional Name" runat = server> .... </HTMLTag> -- here, "Optional Name" is the identifier of the meaning of a server side control object. Although the list, the related syntax, and COM+ class of a HTML tag which are supported now are shown in a table 1, other HTML tags can be considered within the limits of this invention.
[0054]
[A table 1]

HTML タグ名	例	COM+ クラス
<a>	<a id = "MyAnchor" runat = server> My Link </a>	AnchorButton
<img>	<img id = "MyImage" runat = server>	Image
<span>	<span id = "MyLabel" runat = server> </span>	Label
<div>	<div id = "MyDiv" runat = server>Some contents</div>	Panel
<form>	<form id = "MyForm" runat = server> </form>	FormControl
<select>	<select id = "MyList" runat = server> <option>One</option> <option>Two</option> <option>Three</option> </select>	DropDownList
<input type = file>	<input id = "MyFile" type = file runat = server>	FileInput
<input type = text>	<input id = "MyTextBox" type = text>	TextBox
<input type = password>	<input id = "MyPassword" type = password>	TextBox
<input type = reset>	<input id = "MyReset" type = reset>	Button
<input type = radio>	<input id = "MyRadioButton" type = radio runat = server>	RadioButton
<input type = checkbox>	<input id = "MyCheck" type = checkbox runat = server>	CheckBox
<input type = hidden>	<input id ="MyHidden" type = hidden runat = server>	HiddenField
<input type = image>	<input type = image src = "foo.jpg" runat = server>	ImageButton
<input type = submit>	<input type = submit runat = server>	Button
<input type = button>	<input type = button runat = server>	Button
<button>	<button id = MyButton runat = server>	Button
<textarea>	<textarea id = "MyText" runat = server> This is some sample text </textarea>	TextArea

[0055] In addition to a standard HTML control tag, an operation gestalt with this invention enables a developer to create the reusable component encapsulated about a program function common to the outside of a HTML tag set. These custom-made server side control objects are specified using the declaration tag in a page file. Custom-made server side control object declaration includes the "runat" attribute which has the value set as "server". In order to carry out possible [of the program reference of a custom-made control object], the "id" attribute of a meaning is specified as arbitration. Furthermore, the declaration name / value attribute pair of a tab element specify the property argument of a server side control object, and event association. An in-line template parameter may also be combined with a server side control object by giving a suitable "template" prefix child element to a parent server control object. A format of custom-made server side control object declaration is as follows.
[0056] <servercntrlclassname id="OptionalName" [propertyname="propval"] runat=server/> Here, "servercntrlclassname" is an accessible control class, "OptionalName" is the identifier of the meaning of a server side control object, and "propval" expresses the property value of the arbitration of a control object.
[0057] An XML tag prefix can be used for offering the briefer notation for specifying a server side control object in a page by using the following formats using another specification statement method.
[0058] <tagprefix:classname id = "OptionalName" runat = server/> Here, in "tagprefix", in relation to a given control name tooth-space library, "classname" expresses the identifier of control in a relation name tooth-space library. "propertyvalue" of arbitration is supported.
[0059] In short, the operation gestalt of this invention contains on a server creation and the server side

control object by which executive operation is carried out, in order to generate the HTML code sent to a client. The HTML code can embody any effective HTML configurations, for example, is control of an ACTIVEX type and JAVA. Any web resources which generate the user interface carbon button and other user interface elements in an applet, a script, and other clients can be referred to. The user in a client can have a dialog with these user interface elements that correspond to a server side control object logically, and can return a request to a server. A server side control object is re-created on a server, and it processes the data, the event, and other properties of a user interface element so that the HTML code of the next round which should transmit to a client as a response may be generated.
[0060] If drawing 5 is referred to, an example of the computer system of the operation gestalt of this invention contains the general purpose computer equipment of the gestalt of the conventional computer system 500 containing the system bus 506 which connects the various system components containing the processor unit 502, a system memory 504, and a system memory 504 to the processor unit 500. System buses 506 may be any of the bus structure of some types containing the peripheral bus and local bus which use a memory bus or a memory controller, and various bus architectures. The system memory contains the memory (ROM) 508 only for playbacks, and random access memory (RAM) 510. The unformatted input / output system 512 (BIOS) containing the basic routine which helps a transfer of the information between the elements within a computer system 500 are stored in ROM508.
[0061] The computer system 500 contains the optical disk drive 518 which performs read-out and the writing of a removable optical disk 519 like the magnetic disk drive 514 and CD ROM which perform further read-out and the writing of a hard disk drive 512 and the removable magnetic disk 516 which perform read-out and the writing of a hard disk, DVD, or other optical media. The hard disk drive 512, the magnetic disk drive 514, and the optical disk drive 518 are connected by the hard disk drive interface 520, the magnetic disk drive interface 522, and the optical disk drive interface 524 system bus 506, respectively. A drive and its medium which can be related computer read offer the instruction of a computer system 500 which can be computer read, DS, a program, and the non-volatile storage section of other data.
[0062] Although the hard disk, the removable magnetic disk 516, and the removable optical disk 519 are used for this description in the above-mentioned environmental example of a publication, the medium type [other] in which data storage is possible and which can be computer read can be used for the above-mentioned example of a system. The medium of other types of these which can be used for the above-mentioned example of operating environment which can be computer read has for example, a magnetic cassette, flash memory card, a digital videodisc, a BERUNUI (Bernoulli) cartridge, random access memory (RAM), the memory (ROM) only for playbacks, etc.
[0063] Many program modules are stored in a hard disk, a magnetic disk 516, an optical disk 519, and ROM508 or RAM510, and these contain the application program 528, other program modules 530, and the program data 532 of 526 or 1 or more operating systems. A user can input a command and information into a computer system 500 with input units, such as a keyboard 534 and a mouse 536, or other pointing equipments. As other input devices, there are a microphone, a joystick, a gamepad, a satellite dish, a scanner, etc., for example. These and other input devices are connected to the processor 502 through the serial port interface 540 connected to the system bus 506 in many cases. However, these input devices may be connected by other interfaces, such as a parallel port, a game port, or Universal Serial Bus (USB), again. The monitor 542 or the indicating equipment of other types is also connected with the system bus 506 through the interface of a video adapter 544 etc. In addition to a monitor 542, typically, a computer system contains other circumference output units (not shown), such as a loudspeaker and a printer.
[0064] A computer system 500 can operate in the environment using the logical connection to one or more remote computers like a remote computer 546 connected by network. a remote computer 546 -- a computer system, a server, a router, Network PC, and a pier (peer) -- it may be equipment or other common network nodes, and there are many elements typically mentioned above in connection with a computer system 500, or all are included. Network connection contains Local Area Network (LAN) 548 and Wide Area Network (WAN) 550. Such a network environment is not new in office, an enterprise

magnitude computer network, intranet, and the Internet.
[0065] When using by the LAN network environment, a computer system 500 is connected to a local network 548 through a network interface or an adapter 552. When using by the WAN network environment, a computer system 500 includes the modem 554 or other means for establishing the communication link by Wide Area Network 550 like the Internet typically. Built-in or external any is sufficient as a modem 554, and it is connected with the system bus 506 through the serial port interface 540. In the environment connected by network, the program module described in relation to the computer system 500 or its part may be memorized by remote memory storage. The illustrated network connection is an example and can use other means for the communication link establishment between computers.
[0066] In the operation gestalt of this invention, a computer 500 expresses a web server and CPU502 carries out executive operation of the page factory module on the ASP+ resource memorized by at least one of storages 516, 512, 514, 518, and 519 or memory 504. A HTTP response and a request communicate by LAN548 connected to the client computer 546.
[0067] Drawing 6 is a process flowchart showing server side processing of the page object in an operation gestalt with this invention, and other control objects. In processing 600, the page object construction section is called with the page factory module 308 (refer to drawing 3). Consequently, a page object (see the page object 314 in drawing 3) is created so that it may correspond to the web page user interface element and logic target on a client. In processing 602, a page factory module calls the ProcessRequest() member function of the page object which begins to process [gradual] the HTTP request received from the client. In the 1st phase of 1 operation gestalt of this invention, server side creation processing (not shown) is creation of the descendant server side control object contained in the control object hierarchy of a page object. That is, in order that the construction section of a child control object may create the control object between the processing life times of HTTP request processing, call appearance is carried out repeatedly.
[0068] However, in another operation gestalt, creation of a child control object is delayable until a control object is needed for a given processing step (for example, rendering of the HTML code of the user interface element with which handling of a postback event, handling of postback data, loading of a view state and preservation, and data coupling decompose or correspond). Since the operation gestalt of the latter "which delays control object creation" can reduce utilization of unnecessary CPU and memory, it is the optimal. For example, there is a case where it will be called creation of the web page from which the user input event received from the client completely differs. In this case, a control object hierarchy makes it end promptly, and only in order to process the event which will illustrate a new different control object hierarchy of a new page, it is not necessary to illustrate the whole control object hierarchy of a former page.
[0069] The server call of the ProcessRequest method of a page object can be answered, and executive operation of the processings 604-620 can be carried out by the data page object and each descendant control object according to the data of a given HTTP request etc. In an operation gestalt with this invention, processings 604-620 are performed to each object of each in the sequence of drawing 6 . However, depending on a HTTP request, about given processing of another object, it may not be carried out in order or, as for the given processing to one object, processing may not be performed at all. For example, the 1st object performs the initialization processing 604 and its load processing 606, the postback data processing 608 is started and a descendant control object performs the initialization processing 604 and the load processing 606 of itself by overdue control object creation after that. Although the sequence of processing by the page object and the descendant control object is not restricted to this, it depends on the various factors [whether the current condition of the property of the data in a HTTP request, a control object hierarchy's configuration, and a control object and control object creation are performed behind time] to include.
[0070] The initialization processing 604 initializes a control object, after a control object is created by carrying out executive operation of the server side code of the arbitration about initialization in a dynamic contents resource. Thus, each server side control object may be customized by the specific

server side function declared with the dynamic contents resource. In the operation gestalt of this invention, a dynamic contents code customizes or extends the base page control class declared by the page developer with the ASP+ resource on a server. Compile of an ASP+ resource contains the declared code in the suitable initial code (for example, initialization () method of a page object and a descendant control object). The initialization processing 604 carries out executive operation of this code, and customizes or extends a page base class and the base class of a descendant control object.
[0071] In an operation gestalt with this invention, the status management of a server side control object is supported in the load processing 606 and the preservation processing 616 in which the condition structure which can be conveyed is used, in order to contain the non-statement model of a client/server system by returning a server side control object to a former condition. With a certain operation gestalt, although a condition goes and comes back to a server in the one or more HTML fields in which the HTTP request / response of a couple hid, it is considered that other condition structures which can be conveyed are within the limits of this invention.
[0072] In a series of processings by the given present request and present given response about a page between a client and a server, the condition of one or more control objects is recorded on the condition structure which can be conveyed by the preservation processing 616 after processing of a previous request. In an operation gestalt with this invention, the status information of the addition containing the control object identifier which enables a hierarchy or a server to associate a suitable control object and a given condition is also included in the condition structure which can be conveyed. In the next HTTP request, status information is returned to a server with the condition structure which can be conveyed. A server extracts status information from the received condition structure which can be conveyed, loads condition data to the suitable control object in a control object hierarchy, and returns each control object to the condition that it existed before the previous HTTP response. After the processing to the present request, again, the condition of one or more server side control objects is recorded on the condition structure which can be conveyed by the preservation processing 616, and returns the condition structure which can be conveyed to a client by the next HTTP response by it.
[0073] Each server side control object is changed into the same condition as the condition before a previous HTTP request as a result of the load processing 606. For example, when a text box control object includes a property value equivalent to "JDoe" before a previous HTTP response, the load processing 606 returns the same control object to a previous condition by loading text string "JDoe" to the property value etc. Furthermore, whether the condition of a given object was memorized and recovered can also constitute.
[0074] When 1 operation gestalt of this invention is summarized, "it is saved" after the condition of one or more server side control objects processing. The saved status information is transmitted to a client by the response. A client returns the status information saved by the next response to a server side. A server loads status information to the newly illustrated server side control object hierarchy so that a hierarchy's condition may return to a former condition.
[0075] With another operation gestalt, status information can be held for some other accessible web locations by the server during round trip on a server of returning from a server to a client and a server. After a server receives a client request, this status information is taken out by the server and may be loaded to the suitable server side control object in a control object hierarchy.
[0076] In processing 608, the postback data received from the HTTP request are processed. Postback data may be contained in the pay load of the HTTP request by the key value and hierarchical display (for example, XML) which became a pair, or other data display like RDF (Resource Description Framework). Processing 608 analyzes the syntax of a pay load, and identifies the identifier of the meaning of a server side control object. When an identifier (for example, "page1:text1") is found and the identified server side control object exists in a control object hierarchy, corresponding postback data are passed to the control object. For example, reference of drawing 1 sends the identifier of a meaning relevant to a text box 106 and text "JDoe" to a web server 116 with the pay load of the HTTP request 114. Processing 608 analyzes the syntax of the pay load of the HTTP request 114, and acquires the identifier and its related value (namely, "JDoe") of a meaning of a text box 106. And processing 608

decomposes the identifier of the meaning of a text box 106, identifies a corresponding server side control object, and passes a "JDoe" value to the object to process.
[0077] As the load processing 606 was explained, the property value of a server side control object may be returned to a former condition. If postback data are received, a server side control object will judge whether the property value of the point to which the passed postback value corresponds was changed. When it changes, logging of the change is carried out to the change list in which data change of a related control object is shown. After all postback data are processed within a control object hierarchy, a control object method is called and one or more postdata change events can be raised to one or more non-user interface server components like stock price retrieval application started on a server. A postdata change event is an event which shows that for example, postback data changed the property of a server side control object. In an instantiation operation gestalt, such an event is sent to a system offer event queue, and can call the server side code registered so that an event might be processed. And a server side code can call the method of a non-user interface server component. Thus, a server side non-user interface server component can answer the event in which the trigger was carried out by change of the data of a server side control object. The option including using an application offer event queue, polling, and processing interruption which performs an event can also be considered within the limits of this invention.
[0078] A postback event is handled in processing 610. A postback event can communicate with the pay load of a HTTP request. Processing 610 analyzes the syntax of the specific event target (for example, with the operation gestalt with this invention, label attachment is carried out with "##EVENTTARGET") which identifies the server side control object to which the event is turned. Furthermore, processing 610 analyzes the syntax of it, when there is a discovered event argument, and it gives it to the server side control object which had the event argument (for example, label attachment is carried out with "##EVENTARGUMENT" with the operation gestalt with this invention) specified. A control object starts the event processed in server side code which calls the method of the non-user interface server component (for example, server side stock price retrieval application) relevant to a dynamic contents resource.
[0079] Processing 612 solves the data-coupling relation between a server side control object and one or more databases with an accessible server, and updates it in a control object property by this with a database value, and updates the database field with the value of/or a control object property. the property of a server side control object may be related with the property of a parent data-coupling container as shown in the table of a server side application database with an operation gestalt with this invention (or data coupling is carried out -- having). A page frame work piece can update the control object property which has the value of the parent data-coupling container property corresponding to between the data-coupling processings 612 and by which data coupling was carried out. Thus, the user interface element on the web page in the next response reflects the updated property value in accuracy. The control object property with which a user interface element corresponds is because it was automatically updated between the data-coupling processings 612. Similarly, a control object property may be updated by the parent data-coupling container field, and, thereby, updates a server side application database in the postback input from a server side control object again.
[0080] Processing 614 updates a large number by which executive operation may be carried out, before saving a control object condition and carrying out the rendering of the output. Processing 616 requires status information (namely, view state) from one or more control objects in a control object hierarchy, stores status information, and inserts it in the condition structure which is sent to a client with a HTTP response pay load and which can be conveyed. For example, a "grid" control object can save the present index page of the list of values, and a "grid" control object can return to this condition after the next HTTP request (namely, processing 606). As mentioned above, view state information expresses the condition of the control object hierarchy before the next action by the client. It will be used for changing a control object hierarchy into the condition of the point before client post back input process or data coupling if view state information returns.
[0081] The rendering processing 618 generates the suitable authoring language output (for example,

HTML data) sent to a client by the HTTP response. A rendering is performed by top-down hierarchy tree WOKU and the embedded rendering code of all server side control objects. Processing 620 does the last clean-up activity (it is connection of a database in closing a file ****) of arbitration, and a control object hierarchy is terminated. Subsequently, processing performs return and processing 622 to 602, and a page object is ended by calling the destructive section there.
[0082] Drawing 7 expresses the notation of an example of the server side control class in an operation gestalt with this invention. The server side control class defines the method common to all the server side control objects in an operation gestalt with this invention, the property, and the event. The more concrete control class (for example, server side carbon button control object corresponding to the client side carbon button in a web page) is derived from this control class. The shown control class 700 contains the memory which stores a property 702 and a method 704. The operation gestalt of other control classes from which the combination of a data member and a method differs is also considered within the limits of this invention.
[0083] In the shown operation gestalt, the property 702 is public. Property"ID" is a string value in which reading which shows the control object identifier and writing are possible. Property "Visible" is a boolean value in which reading which shows whether the rendering of the authoring language data of a corresponding client side user interface element should be carried out and writing are possible. Property "MaintainState" is a boolean value in which reading which shows whether a control object should save the view state (and view state of the child object) at the time of termination of the present page request (by namely, response to the preservation processing 616 of drawing 6) and writing are possible. Property "Parent" is a reference in which reading by the control container (refer to drawing 8) related in a control object hierarchy's present control object is possible. Property "Page" is a reference in which reading by the root page object to which it acts as the host of the current control object is possible. Property "Form" is a reference in which reading by the form control object to which it acts as the host of the current control object is possible. Property "Trace" is a reference which enables the writing of a developer's trace log and which can be read. A property "BindingContainer" is a reference in which reading by the immediate-data joint container of a control object is possible. Property "Bindings" is a reference in which reading by the set of control object data-coupling association is possible.
[0084] A method 704 includes the access approach to processing of a request, and the data member of a control object. A method is referred in a certain operation gestalt by the pointer stored in the memory tooth space of a control object. This reference means is used in the operation gestalt of ** and a container control object, a control collection object, and other server side objects containing a page object. Method"Init()" is used for initializing these after a child control object is created (see the processing 604 of drawing 6). Method"Load()" is used for recovering view state information from a previous HTTP request (see the processing 606 of drawing 6). Method"Save()" is used for saving the view state information that it uses at a next HTTP request (see the processing 616 of drawing 6). Method"PreRender()" is used for performing a required pre rendering step in advance of preservation of a view state, and the rendering of contents (see the processing 614 of drawing 6). Method"Render (TextWriter output)" is used for outputting the authoring language code of the user interface element corresponding to the present control object (see the processing 618 of drawing 6). Since a code is stored by the response to a client, a code lets an output stream pass and is carried out (it is passed with an "output" parameter). Before method"Dispose()" ends a control object, it is used for doing a final clean-up activity (see the processing 620 of drawing 6).
[0085] Method"GetUniqueID()" obtains the string identifier as which the meaning of a control object was recognized hierarchical. Method"GetControlWithID(String id)" returns the reference to the direct child control object which has the given identifier ("id"). Method"GetControlUniqueWithID(String id)" returns the reference to the child control object which has the hierarchy identifier ("id") of a meaning.
[0086] Method"PushControlPropertyTwoBindingContainer" (String prop Name) will be used for updating the joint container of 2-way data coupling from postback data if a postback data value changes within a server side control object. Method"PushBindingContainerPropertyTwoControl(String prop Name)" is used for updating a server side control object property with a current joint container value.

Method"HasBindings()" returns the boolean value which shows whether the control object has joint association. These three functions are used for solving the joint relation between the property of a server side control object, and the attribute in the server side data storage section (see the data-coupling processing 612 of drawing 6).
[0087] Drawing 8 shows an example of the server side container control class in an operation gestalt with this invention. A container control class supports a nested child control object, and offers serialization and the un-serialized configuration for view state information automatically to the condition structure which can be conveyed. The container control class 800 contains the memory which stores a property 802 and a method 804. Property "Controls" is a reference in which reading by "ControlCollection" of the child control object in a control object hierarchy is possible (see drawing 9). Property "StateCollection" is a reference in which reading by the dictionary of view state information used for maintaining the condition of a control object over much page requests is possible. Method"HasControls()" returns the boolean value which shows whether the control object has the child control object.
[0088] In another operation gestalt, the member property and method of the container control class 800 may be included in the control class 700 of drawing 7 so that each control object can support a child object in itself. However, if such a control object of an operation gestalt does not contain such a child object, a control member (control collection type) will be empty (that is, a child object is not included).
[0089] Drawing 9 shows an example of the server side control collection class in the operation gestalt of this invention. The control collection class 900 contains the memory which stores a property 902 and a method 904. Property "All" is a snap shot dump array in which reading and the writing of all the child control objects of the current control object ordered by the index are possible. Property"this[index]" is a reference in which reading by the control object of the sequence index in a control collection is possible. Property "Count" is a value which shows the number of the child control objects in a collection and which can be read.
[0090] Method"Add'Control child" is used for adding a specific control object to a current collection. Method"IndexOf(Control child)" returns the sequence index of a child control object with which it was specified in the collection. Method"GetEnumerator(boolAllowRemove)" returns ENYUMERETA of all the child control objects in a collection. Method"Remove(Control value)" removes a specific control object from a current collection. Method"Remove(int index)" removes a specific control object from a current collection based on the given index. Method"Clear()" removes all control objects from a current collection.
[0091] Drawing 10 shows an example of the server side page object in an operation gestalt with this invention. A page base class defines the method common to all server side actuation pages, property, and event within the operation gestalt of this invention. The page object 1000 contains the memory which stores the property 1002 and the method 1004. Property "ErrorPage" is a string in whom reading by which a rendering is carried out in the event of an unsettled page exception and writing are possible. Thus, a page object returns the error page which can be read to a client by the HTTP response. Property "Requests" is a reference in which reading by HTTPRequest is possible, and HTTPRequest is given according to the web server framework which gives functionality for close to access the HTTP request data to which it comes.
[0092] Property "Response" is a reference in which reading by the HTTP response given according to the web server framework which offers the function for transmitting HTTP response data to a client is possible. Property "Application" is a reference in which reading by HTTPApplication given according to a web server framework is possible. Property "Session" is a reference in which reading by HTTPSession given according to a web server framework is possible. Property "Server" is a reference in which reading by the server object which is an application server page compatibility utility object is possible. Property "Context" is a reference in which reading of all the web server objects given according to a web server framework is possible, and, thereby, as for a developer, access to the further pipeline module exposure object is attained. property "IsFirstLoad" -- a page object -- for the first time -- or it is the boolean value which shows whether it is loaded and accessed by the response to a client post back

request and which can be read.
[0093] Property"Load()" initializes a page object and is used for recovering view state information from a previous page request. Property"Save()" is used for saving the view state information that it uses for a next page request. Property"HandleError(Exception errorInfo)" is used for treating the unsettled error which takes place between page executive operation. In this event, base class activation returns a client to URL which has a default error web page. Method"GetPostbackScript(Control target, String name, String arg)" returns the client side script method relevant to a given control object. Method"RegisterClientScriptBlock(String key, String script)" is used for eliminating the duplication block of the client side script code transmitted to a client. A duplication block is a script which has the same key value. Method"IHTTPHandler.ProcessRequest(HTTPContext Context)" is used for processing a web request. Since processing of the HTTP request received from the client is initialized, IHTTPHandler.ProcessRequest is called (see the processing 602 of drawing 6). Method"IHTTPHandler.IsReusable()" shows whether a page object can reuse, although much web requests are processed.
[0094] The operation gestalt of this invention given in this description is performed as a logic step of one or more computer systems. Logic processing of this invention is performed as a machine module with which it interconnected within the computer system beyond (2) 1 ** as a processor execute step sequence by which executive operation is carried out by the computer system beyond (1) 1 **. Activation is the problem of selection and it depends for it on the performance requirements of the computer system which performs this invention. Therefore, the logic processing which constitutes the operation gestalt of this invention of a publication on these descriptions can be variously expressed as processing, a step, an object, or a module.
[0095] A description, an above-mentioned example, and above-mentioned data offer the structure of the operation gestalt of this invention, and perfect explanation of an activity. Since this invention can be carried out with many gestalten, without deviating from the pneuma and the range of this invention, this invention is in an attached claim.

[Translation done.]

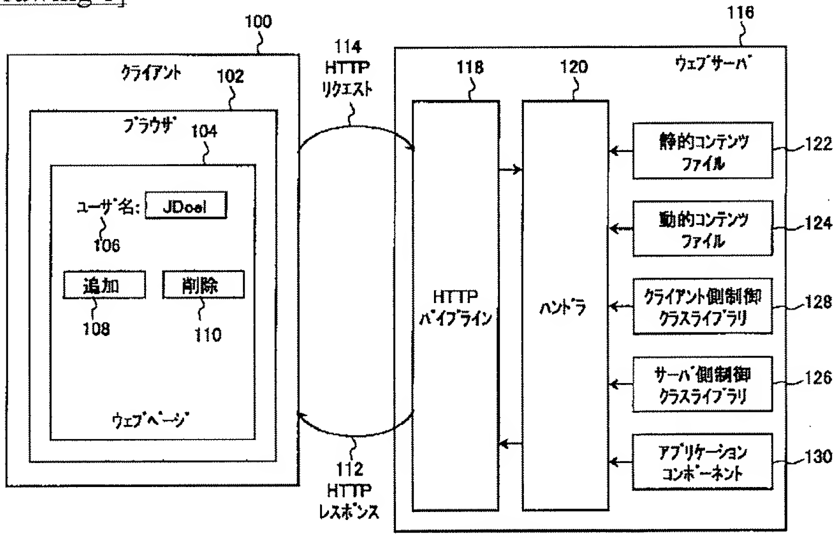
DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] The web server which generates dynamically the web page contents displayed on the client in an operation gestalt with this invention is shown.
[Drawing 2] The flow chart of processing for processing of a client side user interface element and a rendering is shown using the server side control object in an operation gestalt with this invention.
[Drawing 3] An example of the module of a web server used with an operation gestalt with this invention is shown.
[Drawing 4] An example of the dynamic contents resource (for example, ASP+ resource) in an operation gestalt with this invention is shown.
[Drawing 5] An example of a useful system is shown in activation of an operation gestalt with this invention.
[Drawing 6] The flow chart showing processing of the page object in an operation gestalt with this invention is shown.
[Drawing 7] An example of the server side control class in an operation gestalt with this invention is shown.
[Drawing 8] An example of the server side container control class in an operation gestalt with this invention is shown.
[Drawing 9] An example of the server side control collection class in an operation gestalt with this invention is shown.
[Drawing 10] An example of the server side page class in an operation gestalt with this invention is shown.
[Description of Notations]

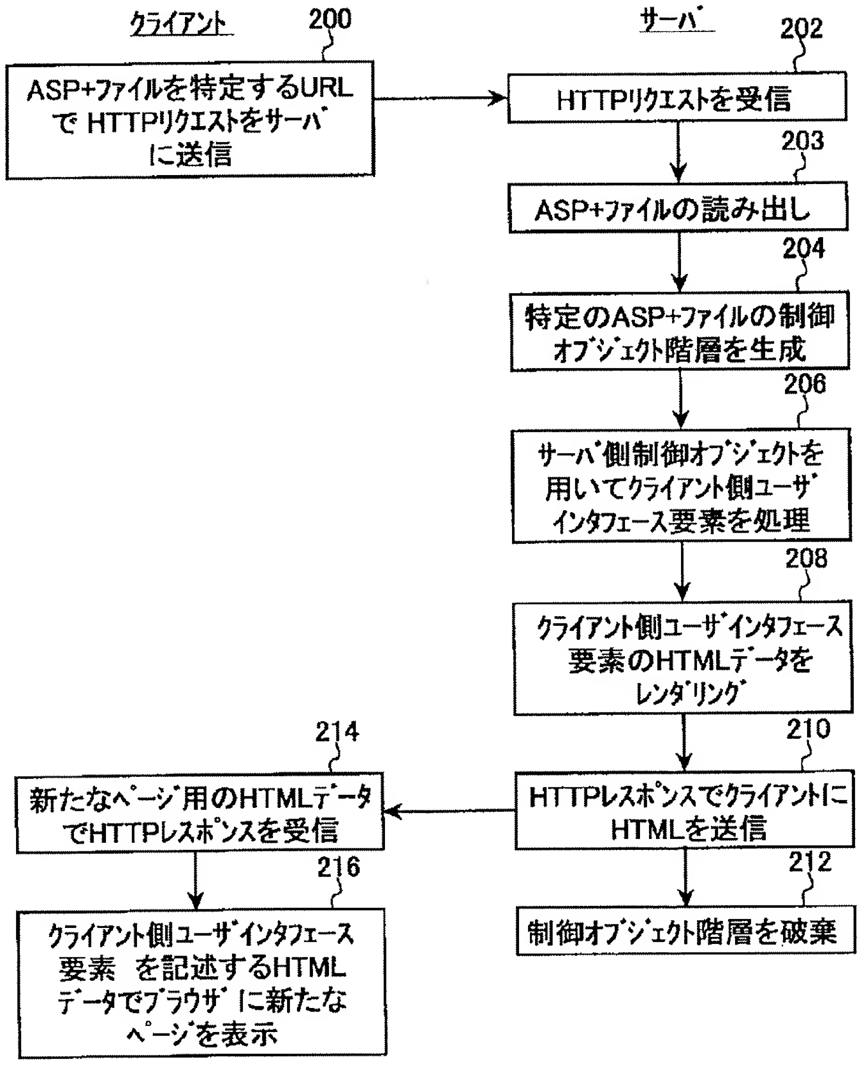
[Translation done.]

DRAWINGS

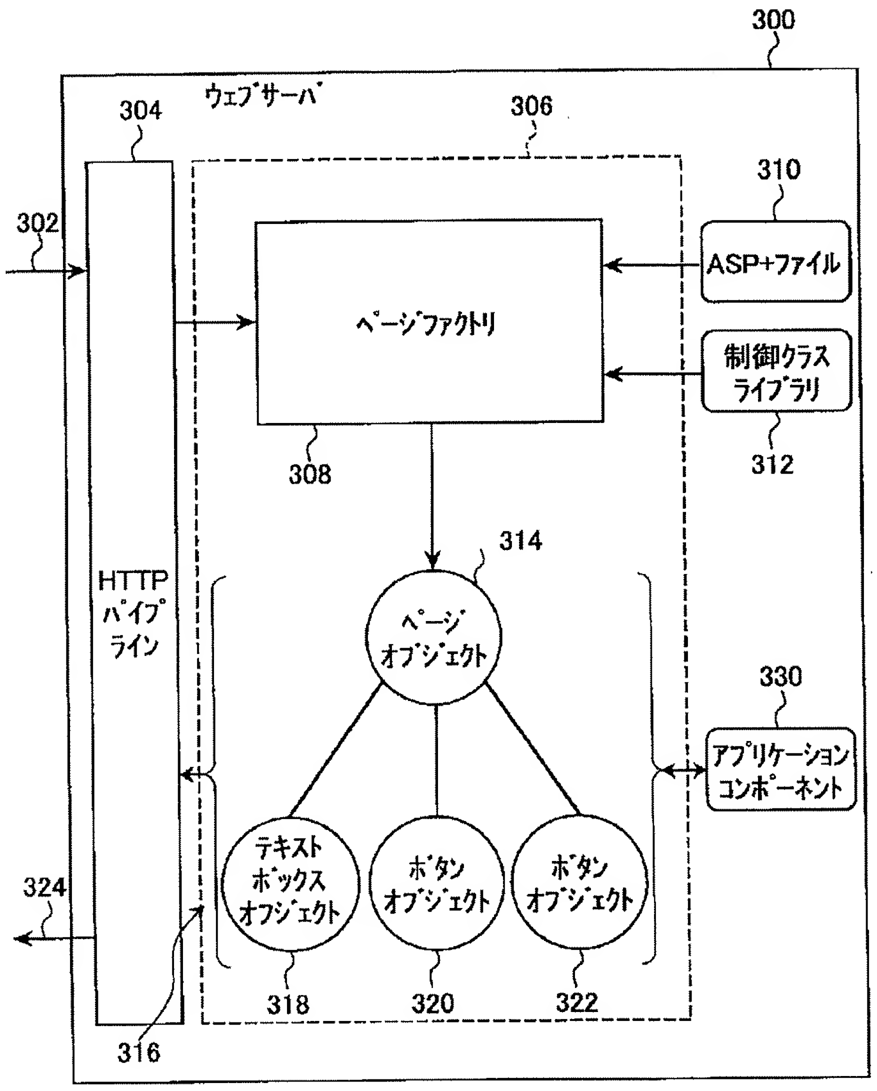
[Drawing 1]



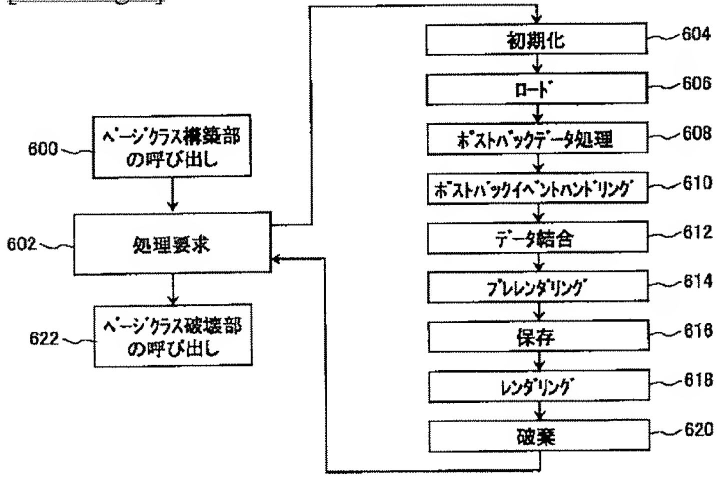
[Drawing 2]


クライアント
200
ASP+ファイルを特定するURLで HTTPリクエストをサーバに送信
サーバ
202
HTTPリクエストを受信
203
ASP+ファイルの読み出し
204
特定のASP+ファイルの制御オブジェクト階層を生成
206
サーバ側制御オブジェクトを用いてクライアント側ユーザインタフェース要素を処理
208
クライアント側ユーザインタフェース要素のHTMLデータをレンダリング
210
HTTPレスポンスでクライアントにHTMLを送信
212
制御オブジェクト階層を破棄
214
新たなページ用のHTMLデータでHTTPレスポンスを受信
216
クライアント側ユーザインタフェース要素 を記述するHTMLデータでブラウザに新たなページを表示

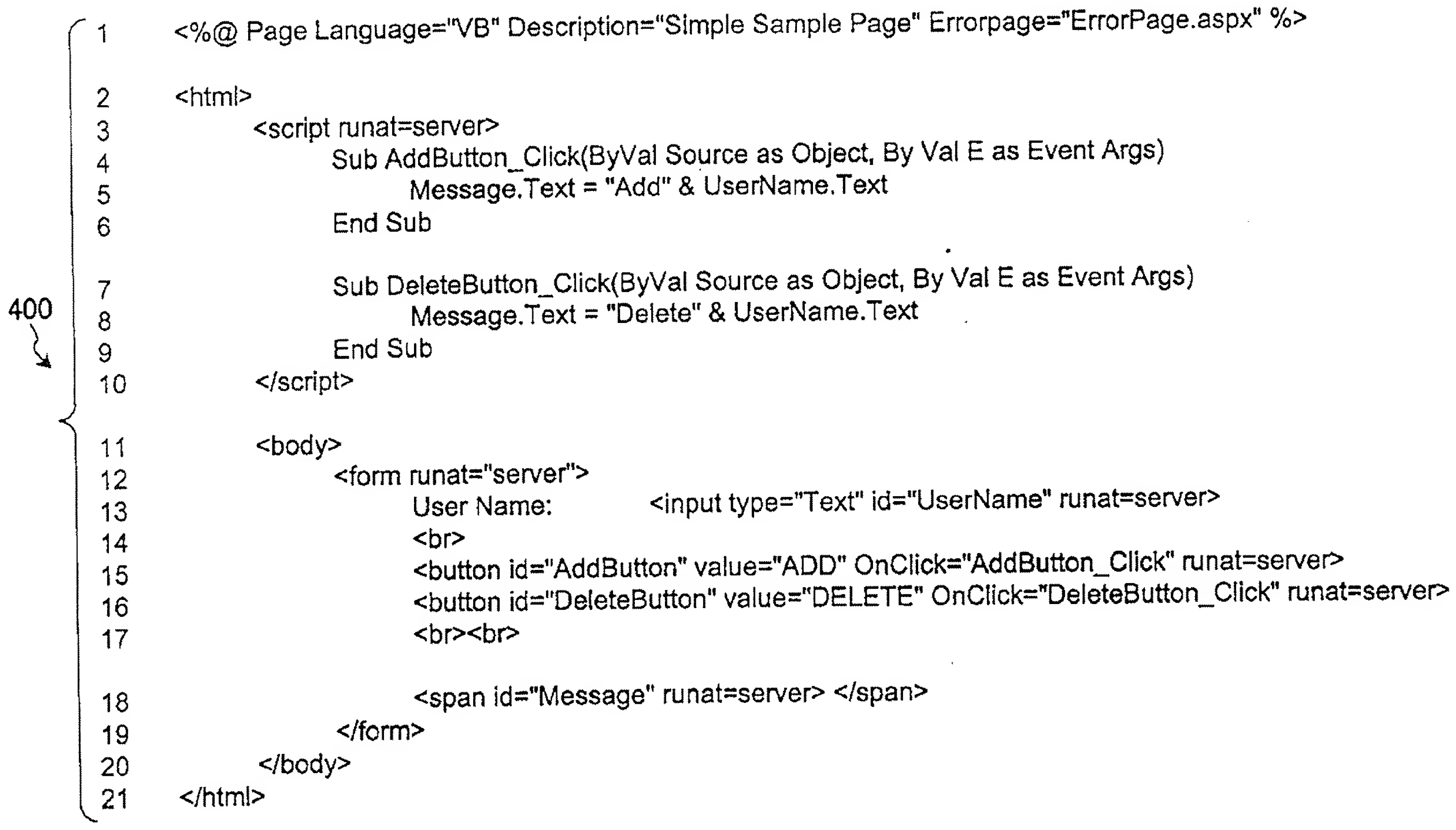
[Drawing 3]



[Drawing 6]

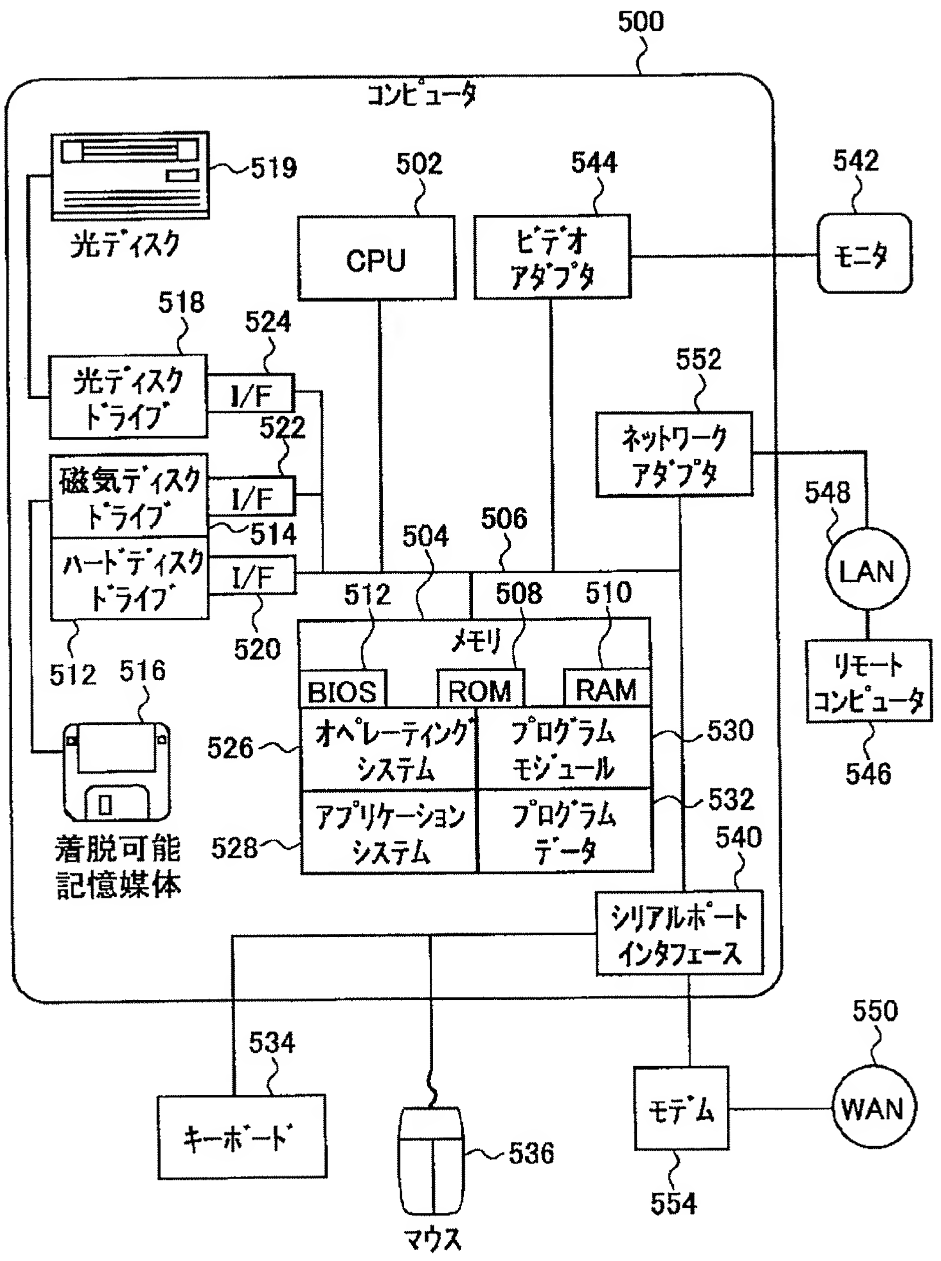


[Drawing 4]

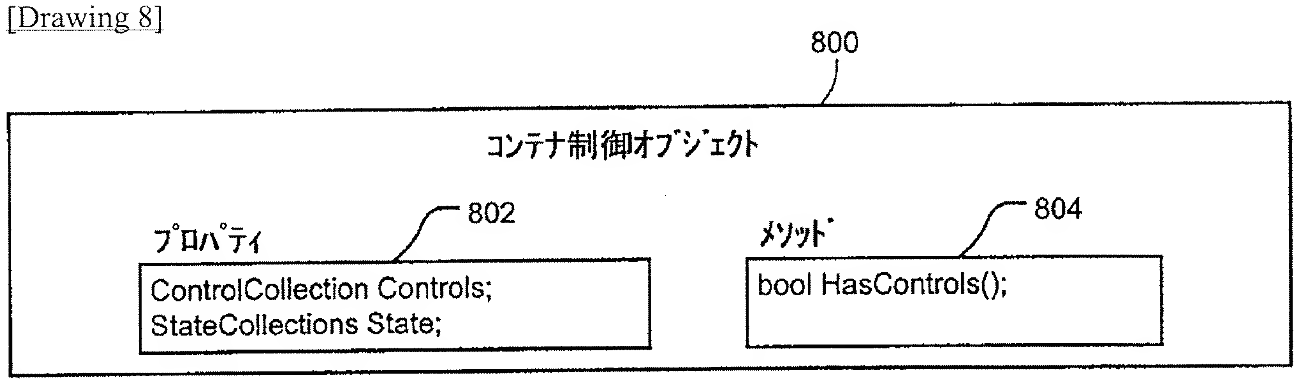


```
<%@ Page Language="VB" Description="Simple Sample Page" Errorpage="ErrorPage.aspx" %>

<html>
          <script runat=server>
                    Sub AddButton_Click(ByVal Source as Object, By Val E as Event Args)
                              Message.Text = "Add" & UserName.Text
                    End Sub

                    Sub DeleteButton_Click(ByVal Source as Object, By Val E as Event Args)
                              Message.Text = "Delete" & UserName.Text
                    End Sub
          </script>

          <body>
                    <form runat="server">
                              User Name:          <input type="Text" id="UserName" runat=server>
                              <br>
                              <button id="AddButton" value="ADD" OnClick="AddButton_Click" runat=server>
                              <button id="DeleteButton" value="DELETE" OnClick="DeleteButton_Click" runat=server>
                              <br><br>

                              <span id="Message" runat=server> </span>
                    </form>
          </body>
</html>
```

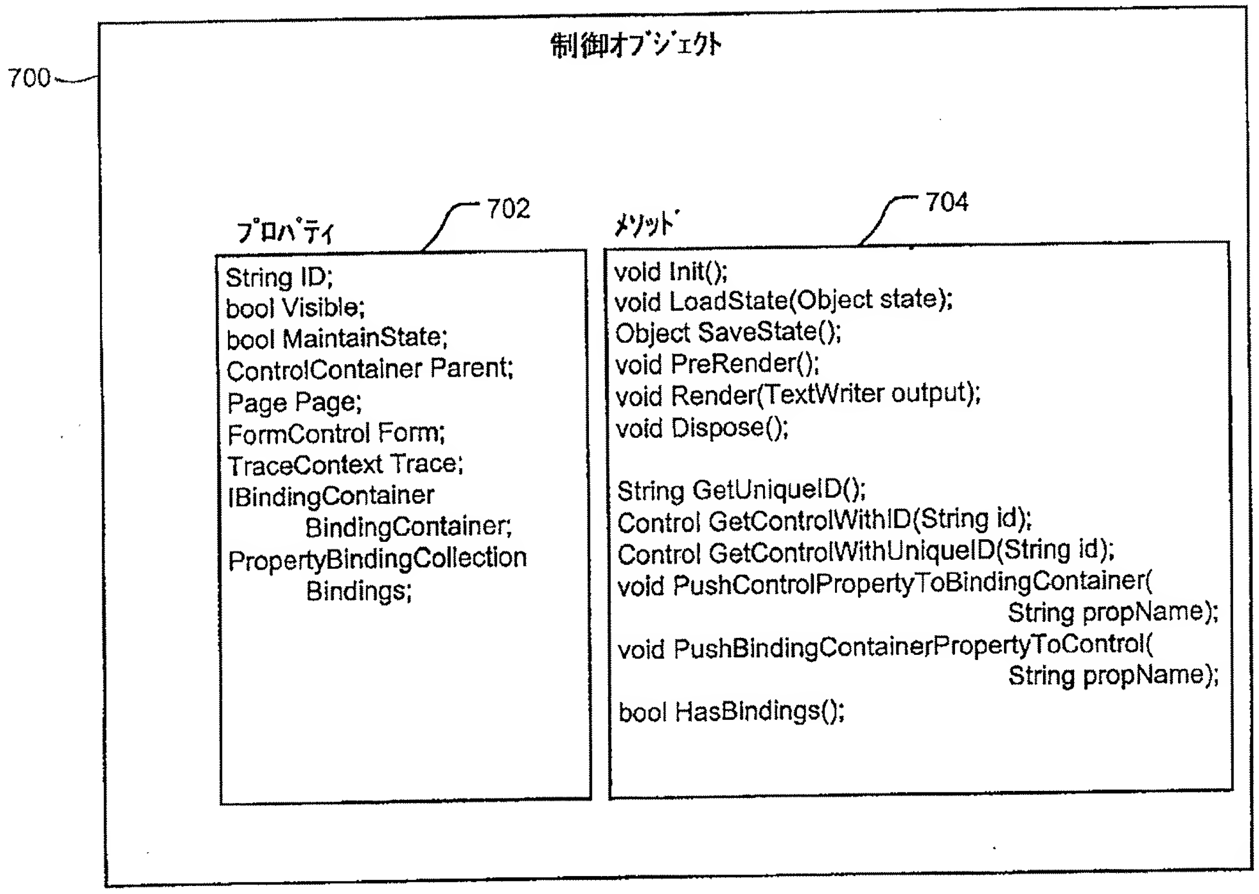
[Drawing 5]



[Drawing 8]



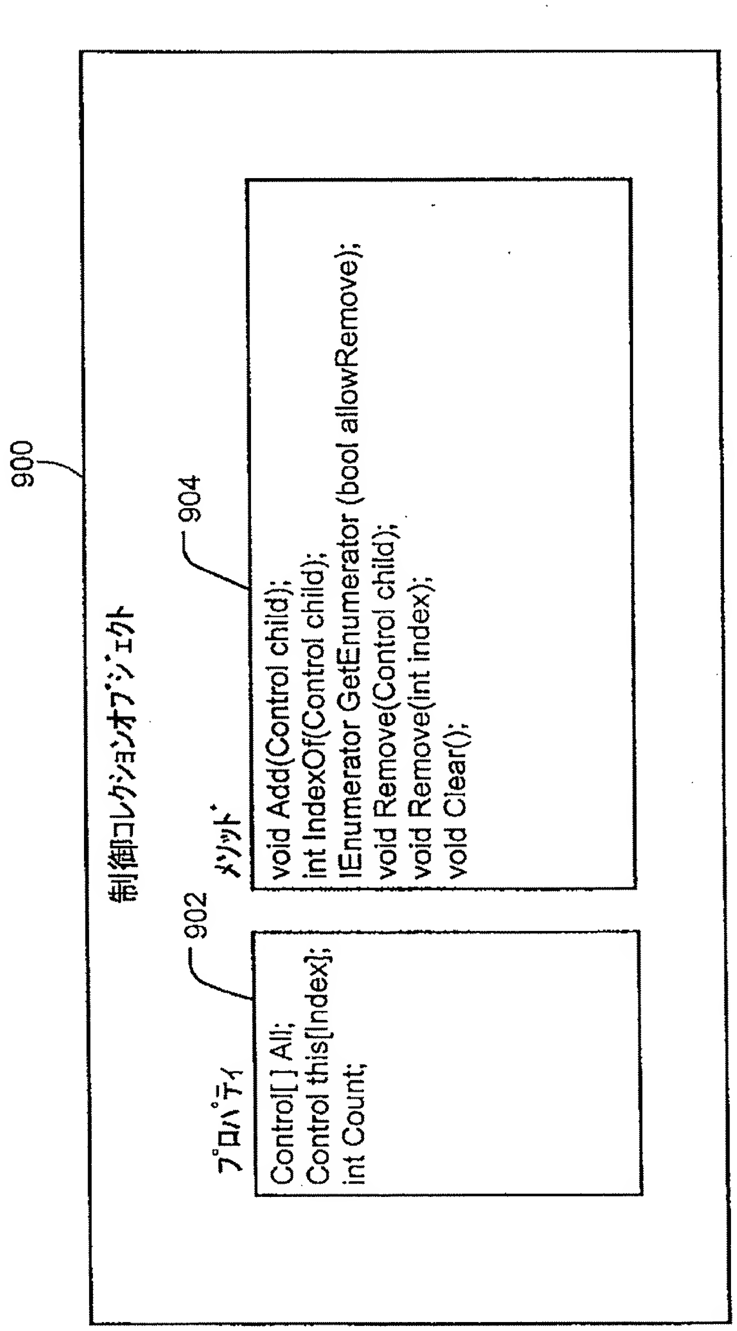
[Drawing 7]



```
String ID;
bool Visible;
bool MaintainState;
ControlContainer Parent;
Page Page;
FormControl Form;
TraceContext Trace;
IBindingContainer
        BindingContainer;
PropertyBindingCollection
        Bindings;
```

```
void Init();
void LoadState(Object state);
Object SaveState();
void PreRender();
void Render(TextWriter output);
void Dispose();

String GetUniqueID();
Control GetControlWithID(String id);
Control GetControlWithUniqueID(String id);
void PushControlPropertyToBindingContainer(
                                String propName);
void PushBindingContainerPropertyToControl(
                                String propName);
bool HasBindings();
```

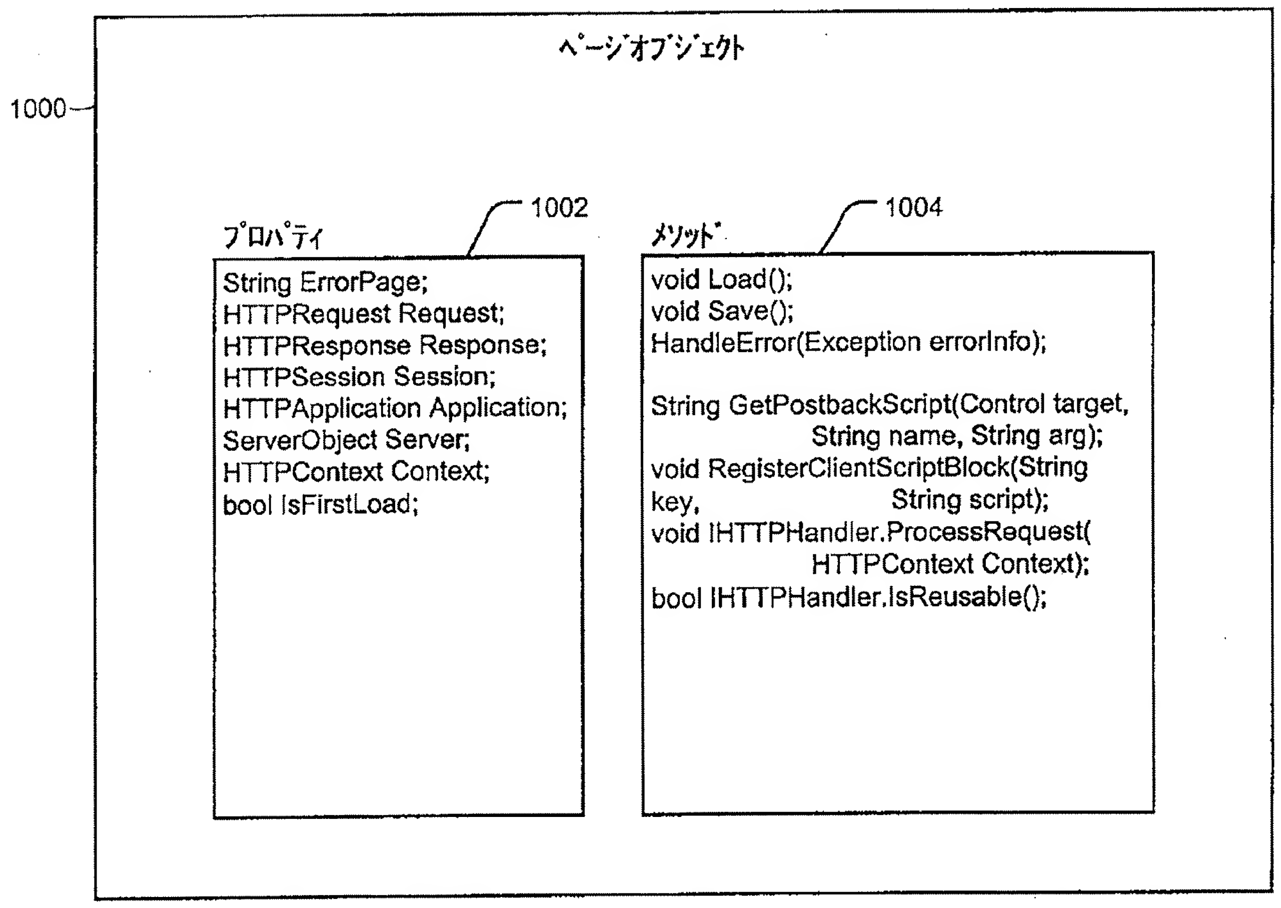
[Drawing 9]



```
Control[] All;
Control this[Index];
int Count;
```

```
void Add(Control child);
int IndexOf(Control child);
IEnumerator GetEnumerator (bool allowRemove);
void Remove(Control child);
void Remove(int index);
void Clear();
```

[Drawing 10]



```
String ErrorPage;
HTTPRequest Request;
HTTPResponse Response;
HTTPSession Session;
HTTPApplication Application;
ServerObject Server;
HTTPContext Context;
bool IsFirstLoad;
```

```
void Load();
void Save();
HandleError(Exception errorInfo);

String GetPostbackScript(Control target,
              String name, String arg);
void RegisterClientScriptBlock(String
key,                  String script);
void IHTTPHandler.ProcessRequest(
              HTTPContext Context);
bool IHTTPHandler.IsReusable();
```

[Translation done.]

WRITTEN AMENDMENT

-- [procedure amendment]
[Filing Date] June 12, Heisei 13 (2001. 6.12)
[Procedure amendment 1]
[Document to be Amended] Description
[Item(s) to be Amended] 0096
[Method of Amendment] Modification
[Proposed Amendment]
[0096]
[Effect of the Invention] Since the server side control object framework which manages processing and generation of a client side user interface element can be offered as mentioned above according to this invention, capsulation of the server side processing about a client side user interface element and a related function can be attained, and it becomes possible to offer the development framework to which a developer can create and process a web page dynamically by the minimum programming.
[Procedure amendment 2]
[Document to be Amended] Description
[Item(s) to be Amended] Explanation of a sign
[Method of Amendment] Modification
[Proposed Amendment]
[Description of Notations]
100 Client
102 Browser
104 Web Page
106 User Name
108 Addition
110 Deletion
112 HTTP Response
114 HTTP Request
116,300 Web server
118 304 HTTP pipeline
120 Handler
122 Static Contents File
124 Dynamic Contents File
126 Server Side Control Class Library
128 Client Side Control Class Library
130 330 Application component
308 Page Factory
310 ASP+ File

312 Control Class Library
314 Page Object
318 Text Box Object
320 322 Carbon button object
500 Computer
504 Memory
512 Hard Disk Drive
514 Magnetic Disk Drive
516 Removable Storage
518 Optical Disk Drive
519 Optical Disk
520, 522, 524 I/F
526 Operating System
528 Application Program
530 Program Module
532 Program Data
534 Keyboard
536 Mouse
540 Serial Port Interface
542 Monitor
546 Remote Computer
552 Network Adaptor
554 Modem
700 Control Object
702, 802, 902, 1002 Property
704, 804, 904, 1004 Method
800 Container Control Object
900 Control Collection Object
1000 Page Object

[Translation done.]